Application Serial No.	:	10/672,390
Filed	:	26 September 2003
Applicant	:	W. Voorhees et al.
Title	:	SYSTEMS AND METHODS FOR CONFIGURING PORTS OF AN SAS DOMAIN
Art Unit	:	2153
Examiner	:	P. J. Chea
Docket Number	:	03-0961
Date	:	30 September 2008

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

Appellants herewith file a Brief in support of their Appeal in the above identified matter. Also being submitted is the $510 fee under 37 CFR 41.20(b)(2) for the Appeal Brief.

TABLE OF CONTENTS

i. REAL PARTY IN INTEREST

The real party in interest is LSI CORPORATION - new corporate name for LSI LOGIC CORPORATION, the employer of the inventor at the time of the invention and the assignee of the patent rights in the above-identified matter. A copy of the corporate resolution changing the name from LSI LOGIC CORPORATION to LSI CORPORATION is attached hereto.

ii. RELATED APPEALS AND INTERFERENCES

No other appeals, interferences, or related applications are known to the Appellants, the Appellants' legal representative, or the Assignee, which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

iii. STATUS OF CLAIMS

Claims 1-12, 14, and 16-20 stand rejected and remain in the application for consideration on appeal. The 35 U.S.C. §103(a) rejection of these claims forms the basis of this appeal.

iv. STATUS OF AMENDMENTS

No amendments have been filed since the last office action (final office action) mailed 9 July 2008.

v. SUMMARY OF THE CLAIMED SUBJECT MATTER

A Serial Attached SCSI (SAS) network (domain) typically comprises one or more SAS initiator devices coupled to one or more SAS target devices via one or more SAS expander devices. In general, as is common in all SCSI communications, SAS initiators initiate communications with SAS targets. The expander devices expand the number of ports of a SAS network domain. The expander devices are often arranged such that the path from any SAS initiator to any particular SAS target may pass through multiple expander devices. In addition, there may exist multiple paths through the network of expanders to establish communications between a particular initiator and a particular target. The expander devices (as well as initiator devices) therefore also include routing tables that enable SAS initiators and SAS devices to route communications through the network of expander devices.

The SAS initiators may also perform as control elements of the network to control and configure the devices of the network for routing information and other attributes. For example, a SAS initiator may require information about a particular SCSI disk drive coupled to an expander device network. The SAS initiator, therefore, sends a SCSI command identifying the desired target device as the ultimate destination for the command. The initiator directs the command to an adjacent expander device of the expander device network. The expander device has routing tables in which it looks for the identified target device and thereby determines the next device along the path to the identified target – perhaps another expander. Each expander similarly consults its routing tables to forward the command further along a path toward the identified target until the intended disk drive target device receives the command. Status information or data generated by the target device to be returned to the initiator is similarly routed through the expander devices until it reaches the appropriate initiator of the SCSI command. Typically, subtractive routing is used to return the response along a path where table routing was used to direct the request. Those skilled in the art will note that table routing is always used in the case of a "fan-out" expander as provided in the SAS specifications.

Prior to commencing normal SAS network domain operations, a topology of the network domain must be identified so that a SAS initiator may correctly configure

routing information to route commands and status through the expander device network. When the SAS network is initialized, a SAS initiator may perform a discovery process that determines the topology of the network domain and configures the routing tables of the expander devices within that domain. Such a discovery process may utilize a Serial Management Protocol (SMP) Report General request and Discover request to determine the topology of the SAS domain (i.e., of the network). SMP protocols and commands are defined in the SAS standards and are used by SAS devices to communicate management information with other SAS devices in a SAS domain. In general, the topology of the network may be determined through the discovery process by a recursive traverse through all expander devices of the SAS domain. An example of such a recursive process is provided in an appendix of the SAS specifications generally available to those of ordinary skill in the art (and publicly available at www.t10.org).

As a preliminary step generally presumed to precede the discovery process, an administrator must set a "routing attribute" associated with each port of a SAS device – in particular the routing attribute needs to be set for each port (also referred to as a Phy) of each expander device. The SAS standards define a "routing attribute" associated with each port that generally indicates the direction of flow through the port as toward target devices or toward initiator devices. "Subtractive routing" as defined in the SAS specification refers to routing between expander devices in a consistent "non-changing" direction through a SAS network domain, such as all routing between expander devices flowing towards a SAS initiator. "Table routing" as defined in the SAS specification refers to routing between expander devices in an opposite direction of subtractive routing. "Direct routing" attribute indicates that the port is coupled directly to a port of an end device (an initiator or target device).

As presently known in the art, configuration of routing attributes of expander device ports or Phys is a manual process performed prior to the discovery process described above for discovering the topology of the SAS domain. When a new SAS domain is initially installed or when an existing SAS domain is altered by addition or subtraction of one or more expanders or other devices, the routing attributes of all ports of all expanders may require re-configuration. While SAS initiators and target devices

typically do not require port route attribute configuration as they are able to configure their respective ports (they are always direct routing), expander devices presently require manual port configuration to associate the subtractive routing attributes and the table routing attributes with ports of the expanders. Since there can be as many as 128 ports or Phys in each SAS device and virtually any number of SAS devices within a single network domain, manually configuring the ports is a tedious and time consuming process and delays operation of the storage network. Further, manual processes for configuring the routing attributes of each port of each device are prone to human errors.

The present invention provides methods and associated structures to automatically configure routing attributes of ports of an SAS network domain. More specifically, one or more domain control elements automatically configure SAS routing attributes of ports for a plurality of expander devices within an SAS network domain such that routing tables of the expander devices can then be automatically formatted. The automated configuration of routing attributes eliminates potential for human error inherent in previous manual processes and reduces delays as compared to such manual processes. The automated methods and structures of the present invention may be embodied within a suitably enhanced SAS initiator device or a suitably enhanced SAS expander of the domain - either operating as a control element.

More specifically, an exemplary embodiment of the invention of claim 1 provides for an automated method of configuring routing attributes of ports within a SAS domain. The method includes automatically discovering devices of the SAS network domain and automatically discovering the ports of the discovered devices (250, 200, 202 of FIG. 2 and page 12 starting at line 26). The method then includes (either following discover processing or integrated with discovery processing) automatically determining and configuring the routing attribute to be associated with each discovered port of the discovered devices (200 of FIG. 2, page 12 starting at line 6; 326 of FIG. 3, page 17 starting at line 12). The method of claim 1 also includes automatically configuring routing table information used by the devices of the domain derived from the configured routing attributes (202 of FIG. 2, page 12, starting at line 11; 320 of FIG. 3, page 16 starting at line 14).

Another exemplary embodiment of the invention as recited in claim 10 provides a SAS network domain comprising a plurality of expander devices each providing a plurality of ports within the domain wherein each port may have an associated routing attribute (102 of FIG. 1, page 7 starting at line 15; E1, E2, and E3 of FIG. 4, page 18 starting at line 11). The SAS domain also includes a domain control element (101 of FIG. 1, page 8 starting at line 4, page 9, starting at line 1 and at line 28) coupled to at least one of the plurality of expander devices operable to configure the routing attributes of the plurality of ports. The domain control element is operable to automatically determine and automatically configure the routing attributes of the ports by traversing port connections between the expander devices. The domain control element is further operable to use the configured routing attributes to automatically generate complete routing tables used by the plurality of expander devices.

Claim 17 reflects yet another exemplary embodiment of the invention. Claim 17 recites means for performing the steps of method claim 1. The structures for performing these functions are the elements of FIG. 1 as discussed above with respect to claim 10.

vi. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Whether claims 1-12, 14, and 16-20 are unpatentable under 35 U.S.C. §103(a) over Applicants Admitted Prior Art (AAPA) in view of Aguilar et al (US 6,199,137).

vii. ARGUMENT

The Examiner rejected all remaining claims (1-12, 14, and 16-20) as unpatentable over AAPA in view of Aguilar. As regards independent claim 1, the Examiner states in essence that AAPA teaches that all the recited steps are known to be manually performed but fails to teach that such steps may be performed automatically. The Examiner then suggests that Aguilar teaches the disadvantages of manual configuration of an IO device (column 1, lines 39-50). The Examiner further suggests that Aguilar teaches automatically discovering devices and ports, and teaches automatically configuring routing table attributes (column 4, lines 26-49).

Applicants respectfully disagree. First and foremost, Aguilar is not related to the particulars of discovering ports of a SAS domain and automatic determination and configuration of routing attributes of devices in the SAS domain. The routing attributes (as described in the subject application) include directional information about the desired use of a port in the domain (i.e., "subtractive", "table", and "direct" routing attribute values). Determination of these attribute values in prior manual techniques required a human thought process to determine the desired routing for each port. The automated configuring of routing attributes as recited in claim 1 requires no such human input to determine the proper routing attribute values for a port. Rather, the proper value is determined and configured automatically by a SAS device (e.g., a "SAS controller" or "SAS control element") in the SAS domain.

AAPA does not suggest that such a determination of routing attribute values may be automated. Neither does Aguilar teach or reasonably suggest any automated technique to replace such a human determination of the proper routing attribute for a port of a SAS device.

"Routing attributes" are a well defined term in the art of SAS interfaces and protocols. Applicants readily admit that it is generally known (as exhibited by Aguilar) to automate the determination and configuration of some parameters of some devices. But such procedures teach nothing of the specifics required for determining and configuring "routing attributes" as the term is well understood by those of ordinary skill in the art of

SAS devices and protocols. Claim 1 is specifically focused on this novel, non-obvious aspect of automatically determining and configuring "routing attributes" of devices in a SAS domain. The SAS specifications suggest no automated methods of so determining the routing attributes of the expander devices in the SAS domain. Aguilar speaks only in the most general sense about automated configuration of certain parameters of certain types of devices - unrelated to the details of "routing attributes" as specific to SAS domains.

The specification and other claims further clarify that the automated process may include traversing nodes of the SAS domain to determine and configure the routing attributes of each port encountered in the traversal.

The Examiner's response to arguments previously presented notes that Aguilar discusses certain automated configurations of a wide variety of devices and then suggests that:

> Given the wide range of domains that could use the benefit of automatic determination and configuration of routing attributes, the Examiner believes that one of ordinary skill in the art at the time of the invention would have found it obvious to apply Aguilar's teaching of automatic determination and configuration of routing attributes to the Applicants Admitted Prior Art (AAPA) of a SAS domain, in order to avoid tedious manual configuration with the addition of new devices (see Aguilar column 1. lines 39-50).

The Examiner applies the teachings of Aguilar that addressed automated configuration of certain devices (not even networks of devices) to the claimed features of configuring routing attributes of devices in a SAS domain. Nothing in Aguilar speaks to configuration of devices in a SAS domain - a network collection of initiator, target, and expander devices where nearly infinite configurations are possible based on routing attributes of the collection of SAS devices. Routing attributes (as the term is properly understood in the context of claim 1 and SAS domains) are nowhere discussed in Aguilar. The automated configuration features of Aguilar (to whatever extent they are expressed) are inapplicable to the complexity of configuring SAS domains. The richness and complexity of the claimed automated procedures is clear in reference to the complexity of the flowchart of FIG. 3. None of the teachings of Aguilar teach or

reasonably suggest the details of such automated determination and configuration of routing attributes in a SAS domain.

Applicants previously amended claim 1 to clearly recite that the routing attribute values are automatically determined and then configured (also automatically). The Examiner's remarks dismiss this aspect suggesting

> In considering the specific routing attributes that are supposed to be configured (i.e. subtractive, table and direct routing attributes), it is not clearly claimed that the routing attributes are limited to those particular ones and even if they were, AAPA discloses that the SAS standards define routing attributes so it would be obvious to configure the ports in that manner.

Again, "routing attributes" in a SAS domain (as clearly recited in claim 1) may only be reasonably understood to mean "routing attributes" as understood in the SAS specification - well known to those of ordinary skill in the art. Applicants readily admit that these attribute values are well known. Indeed they are defined by the SAS specification. However, whatever automated configurations are suggested by Aguilar, none of them suggest techniques that would apply to automated determination and configuration of SAS routing attributes. AAPA certainly suggests that manual steps for such configuration are well known but suggests no techniques for automating those manual steps.

Further, claim 1 recites that the automatically configured routing tables are used by the devices of the SAS domain. In other words, the automatically configured routing tables are distributed or otherwise shared for use by other devices (e.g., all devices) of the SAS domain. Nothing in AAPA or Aguilar teaches or reasonably suggests the automated sharing or distribution of the automatically configured routing tables such that other devices (e.g., all devices) of the SAS domain share the same routing table information. To the contrary, Aguilar appears to discuss the configuration of a single routing device - not the configuration of routing tables that are then used by other devices (e.g., all devices) of a related domain.

Again, in the Examiner's remarks, this feature of using the routing tables that are automatically configured is dismissed suggesting the claim does not recite such. Claim 1

clearly includes the recitation "automatically configuring routing table information used by the devices of the domain derived from the configured routing attributes". The information automatically configured is recited to be "used" by the devices of the domain. Applicants can imagine no clearer recitation than that the automatically configured information is "used" to indicate that all devices of the domain use the information automatically generated in the routing tables. Inherent in such "use" must be distribution or other forms of sharing the automatically configured routing tables.

Thus nothing in AAPA or Aguilar (or any art of record), considered individually or in any combination, teaches or reasonably suggests the features of claim 1 wherein SAS routing attributes are automatically determined and automatically configured to automatically generate routing tables that are used by multiple devices of the SAS domain. Applicants therefore maintain that claim 1 is allowable over all art of record.

There is no reasonable expectation of success in the suggested modification of AAPA by the teachings of Aguilar. Aguilar's proposed automated configurations are simply inapplicable to the complexities of automated determination and configuration of "routing attributes" as the term is properly understood in relation to SAS standards.

Independent claims 10 and 14 include similar recitations and were rejected for similar reasons. Applicants thus maintain that claims 10 and 14 are allowable for at least the same reasons as discussed above with respect to claim 1. Dependent claims 2-9, 11-12, and 16-20 include additional limitations and thus are maintained to be allowable for at least the same reasons as discussed above and as dependent from allowable base claims. In view of the above discussion, Applicants respectfully request reversal of the rejection of all claims under §103 and passage of the application to allowance.

viii. CLAIMS APPENDIX

1. An automated method of configuring routing attributes of ports within a SAS network domain, comprising:

automatically discovering devices of the SAS network domain;

automatically discovering the ports of the discovered devices;

automatically determining the routing attribute to be associated with each discovered port of the discovered devices;

automatically configuring the routing attributes of the discovered ports; and

automatically configuring routing table information used by the devices of the domain derived from the configured routing attributes.

2. The method of claim 1 wherein the steps of discovering devices, discovering ports, and configuring the routing attributes of the discovered ports each include a step of exchanging SMP messages.

3. The method of claim 2 wherein the step of configuring routing table information further comprises:

configuring routing table information within initiator and expander devices of said devices of the SAS network domain wherein said routing table information is sufficient to identify paths in the SAS network domain to enable the exchange of said SMP messages.

4. The method of claim 2 wherein the step of configuring routing table information further comprises:

completely configuring routing table information to identify all paths for exchange of messages within the SAS network domain.

5. The method of claim 4 wherein the step of completely configuring is integrated with the steps of discovering devices, discovering ports, and configuring ports.

6. The method of claim 1 wherein the step of discovering said devices further comprises:

transmitting an SMP Discover request from a first device to a neighboring device of the first device; and

receiving an SMP Discover response in said first device from said neighboring device identifying other devices coupled to ports of said neighboring device.

7. The method of claim 1 wherein the step of discovering said ports of said discovered devices further comprises:

transmitting an SMP Report General request from a first device to a neighboring device of the first device; and

receiving an SMP Report General response in said first device from said neighboring device identifying the number of said ports within said neighboring device.

8. The method of claim 1 wherein the step of configuring further comprises:

transmitting an SMP request from a first device to a second device wherein the SMP request includes vendor unique information identifying a routing attribute of said routing attributes to be configured for a port of said ports of said second device.

9. The method of claim 1 further comprising:

recursively repeating the steps of the method to traverse devices of the SAS network domain to configure said routing attributes of said ports of said devices of the SAS network domain.

10. A SAS network domain, comprising:

a plurality of expander devices providing a plurality of ports within the domain wherein each port may have an associated routing attribute; and

a domain control element coupled to at least one of the plurality of expander devices operable to configure the routing attributes of the plurality of ports, wherein the domain control element is operable to automatically determine and automatically configure the routing attributes of the ports by traversing port connections between the expander devices and wherein the domain control element is further operable to use the configured routing attributes to automatically generate complete routing tables used by the plurality of expander devices.

11. The SAS network domain of claim 10 wherein the domain control element comprises:

a SAS initiator device coupled to at least one of the plurality of expander devices.

12. The SAS network domain of claim 10 wherein the domain control element comprises:

a SAS expander device coupled to at least one of the plurality of expander devices.

13. (Cancelled)

14. A SAS network domain comprising:

means for discovering the topology of the SAS network domain by traversing port connections between devices of the domain;

means for automatically determining the routing attribute to be associated with each discovered port of the discovered devices;

means for configuring SAS routing attributes associated with ports of devices of the domain in response to discovery of the topology of the domain; and

means for configuring routing tables using the configured routing attributes, the routing tables used by the devices of the domain.

15. (Cancelled)

16. The SAS network domain of claim 14 wherein said means for configuring routing tables, said means for discovering and said means for configuring SAS routing attributes are integrated so as to traverse the port connection between the devices of the domain only once.

17. The SAS network domain of claim 14 wherein the means for discovering the topology further comprises:

means for exchanging SMP messages between the devices of the domain to identify the devices, to identify the ports of the devices and to identify the port connections between the ports of the devices.

18. The SAS network domain of claim 17 wherein the means for exchanging SMP messages further comprises:

means for exchanging SMP Report General request and response messages to identify ports of devices and to identify the port connections between the ports of the devices.

19. The SAS network domain of claim 17 wherein the means for exchanging SMP messages further comprises:

means for exchanging SMP Discover request and response messages between the devices of the domain.

20. The SAS network domain of claim 14 wherein the means for configuring further comprises:

means for transmitting an SMP message having vendor unique information from a first device to a second device to instruct the second device to configure the routing attribute of a port of the second device.

xi. EVIDENCE APPENDIX

Included are copies of the evidence relied upon by the Examiner as to the grounds of the rejections under 35 U.S.C. §103(a) to be reviewed on appeal.

1. Subject application as originally filed (cited by the Examiner as Applicants Admitted Prior Art). (34 sheets)

2. Aguilar et al. (United States Patent 6,199,137). (10 sheets)

3. Evidence of change of name of party in interest from LSI Logic to LSI Corporation. (5 sheets)

x. RELATED PROCEEDINGS APPENDIX

None.

SUMMARY

Appellants argue that the Examiner's rejections of claims 1-12, 14, and 16-20 under 35 U.S.C. §103(a) are inadequate as a matter of law and should be reversed. It is believed that this Appeal Brief has been timely filed. However, if an extension of time is deemed to be required by the Patent Office, the Patent Office is hereby requested to accept this request as a petition for an appropriate extension of time to respond with any requisite fees therefore being charged to deposit account 12-2252.

Respectfully submitted,

/Daniel N. Fishman/
Daniel N. Fishman #35,512
Duft, Bornsen & Fishman, LLP
1526 Spruce St.
Suite 302
Boulder, CO 80302
(303) 786-7687
(303) 786-7691 (fax)

SYSTEMS AND METHODS FOR CONFIGURING PORTS OF AN SAS DOMAIN

Background of the Invention

1. Field of the Invention

The invention generally relates to configuring ports of a storage domain. More specifically, the invention relates to configuring ports within a Serial Attached Small Computer Systems Interface ("SAS") domain to operate according to certain routing attributes.

2. Discussion of Related Art

Small Computer Systems Interface ("SCSI") is a set of American National Standards Institute ("ANSI") standard electronic interface specification that allow, for example, computers to communicate with peripheral hardware. Common SCSI compatible peripheral devices may include: disk drives, tape drives, Compact Disc-Read Only Memory ("CD-ROM") drives, printers and scanners. SCSI as originally created included both a command/response data structure specification and an interface and protocol standard for a parallel bus structure for attachment of devices. SCSI has evolved from exclusively parallel interfaces to include both parallel and serial interfaces. SCSI now refers to a plurality of primary commands common to most devices and command sets to meet the needs of specific device types as well as a variety of interface standards and protocols.

The collection of primary commands and other command sets may be used with SCSI parallel interfaces as well as with serial interfaces. The serial interface standards that support SCSI command processing include: Fibre Channel, Serial Bus Protocol (used with the Institute of Electrical and Electronics Engineers 1394 FireWire physical protocol; "IEEE 1394") and the Serial Storage Protocol (SSP). SCSI interfaces and commands are also used to network storage devices with processing devices having serial interfaces such as Serial Attached SCSI ("SAS") and Serial Advanced Technology Attachment ("SATA"). These applications are often referred to as storage networks. Such SCSI storage networks are often used in large storage systems having a plurality of disk drives to store data for organizations and/or businesses. The network architecture allows storage devices to be physically dispersed in an enterprise while continuing to directly support SCSI commands

directly. This architecture allows for distribution of the storage components in an enterprise without the need for added overhead in converting storage requests from SCSI commands into other network commands and then back into lower level SCSI storage related commands. Those skilled in the art are familiar with SAS and SATA standards as well as other SCSI related specifications and standards. Information about such interfaces and commands is generally obtainable at the website http://www.t10.org.

An SAS network typically comprises one or more SAS initiators coupled to one or more SAS targets via one or more SAS expander devices. In general, as is common in all SCSI communications, SAS initiators initiate communications with SAS targets. The expander devices expand the number of ports of an SAS network domain. The expander devices are often arranged such that the path from any SAS initiator to any particular SAS target may pass through multiple expander devices. In addition, there may exist multiple paths through the network of expanders to establish communications between a particular initiator and a particular target. The expander devices (as well as initiator devices) therefore also include routing tables that enable SAS initiators and SAS devices to route communications through the network of expander devices.

The SAS initiators may also perform as control elements of the network to control and configure the devices of the network for routing information and other attributes. For example, an SAS initiator may require information about a particular SCSI disk drive coupled to an expander device network. The SAS initiator, therefore, sends a SCSI command identifying the desired target device as the ultimate destination for the command. The initiator directs the command to an adjacent expander device of the expander device network. The expander device has routing tables in which it looks for the identified target device and thereby determines the next device along the path to the identified target – perhaps another expander. Each expander similarly consults its routing tables to forward the command further along a path toward the identified target until the intended disk drive target device receives the command. Status information or data generated by the target device to be returned to the initiator is similarly routed through the expander devices until it reaches the appropriate initiator of the SCSI command. Typically, subtractive routing is used to

return the response along a path where table routing was used to direct the request. Those skilled in the art will note that table routing is always used in the case of a "fan-out" expander as provided in the SAS specifications.

Prior to commencing normal SAS network domain operations, a topology of the network domain must be identified so that an SAS initiator may correctly configure routing information to route commands and status through the expander device network. When the SAS network is initialized, an SAS initiator may perform a discovery process that determines the topology of the network domain and configures the routing tables of the expander devices within that domain. Such a discovery process may utilize a Serial Management Protocol ("SMP") Report General request and Discover request to determine the topology of the SAS domain (i.e., of the network). SMP protocols and commands are defined in the SAS standards and are used by SAS devices to communicate management information with other SAS devices in an SAS domain. In general, the topology of the network may be determined through the discovery process by a recursive traverse through all expander devices of the SAS domain. An example of such a recursive process is provided in an appendix of the SAS specifications generally available at the web site noted above.

As a preliminary step generally presumed to precede the discovery process, an administrator must set a "routing attribute" associated with each port of an SAS device – in particular the routing attribute needs to be set for each port (also referred to as a Phy) of each expander device. The SAS standards define a "routing attribute" associated with each port that generally indicates the direction of flow through the port as toward target devices or toward initiator devices. "Subtractive routing" as defined in the SAS specification refers to routing between expander devices in a consistent "non-changing" direction through an SAS network domain, such as all routing between expander devices flowing towards an SAS initiator. "Table routing" as defined in the SAS specification refers to routing between expander devices in an opposite direction of subtractive routing. "Direct routing" attribute indicates that the port is coupled directly to a port of an end device (an initiator or target device).

As presently known in the art, configuration of routing attributes of expander device ports or Phys is a manual process performed prior to the discovery process described above for discovering the topology of the SAS domain. When a new SAS

domain is initially installed or when an existing SAS domain is altered by addition or subtraction of one or more expanders or other devices, the routing attributes of all ports of all expanders may require re-configuration. While SAS initiators and target devices typically do not require port route attribute configuration as they are able to configure their respective ports (they are always direct routing), expander devices presently require manual port configuration to associate the subtractive routing attributes and the table routing attributes with ports of the expanders. Since there can be as many as 128 ports or Phys in each SAS device and virtually any number of SAS devices within a single network domain, manually configuring the ports is a tedious and time consuming process and delays operation of the storage network. Further, manual processes for configuring the routing attributes of each port of each device are prone to human errors.

In view of the above discussion, it is evident that there is a need for improved systems and methods for configuring routing attributes of ports of an SAS network domain.

Summary of the Invention

The present invention solves the above and other problems, thereby advancing the state of useful arts, by providing methods and associated structures to automatically configure routing attributes of ports of an SAS network domain. More specifically, one or more domain control elements automatically configure SAS routing attributes of ports for a plurality of expander devices within an SAS network domain such that routing tables of the expander devices can then be formatted. In one aspect hereof, a domain control element is communicatively coupled to the expander devices. The control element traverses the ports in each of the expanders of the domain setting routing attributes of ports as it proceeds. The automated configuration of routing attributes eliminates potential for human error inherent in previous manual processes and reduces delays as compared to such manual processes. In one aspect hereof, the control element may be a SAS initiator device. In another aspect, the control element may be a SAS expander device.

One feature hereof therefore provides an automated method of configuring routing attributes of ports within an SAS network domain, comprising: automatically discovering devices of the SAS network domain; automatically discovering ports of

the discovered devices; and automatically configuring routing attributes of discovered ports.

Another aspect hereof further provides that the steps of discovering devices, discovering ports and configuring ports each include a step of exchanging SMP messages.

Another aspect hereof further provides for configuring routing table information within devices of the SAS network domain wherein said routing table information is sufficient to identify paths in the SAS network domain to enable the exchange of said SMP messages.

Another aspect hereof further provides for completely configuring routing table information to identify all paths for exchange of messages within the SAS network domain.

Another aspect hereof further provides that the step of completely configuring is substantially integrated with the steps of discovering devices, discovering ports and configuring ports .

Another aspect hereof further provides that the step of discovering devices further comprises: transmitting an SMP Discover request from a first device to a neighboring device of the first device; and receiving an SMP Discover response in said first device from said neighboring device identifying the other devices coupled to ports of said neighboring device.

Another aspect hereof further provides that the step of discovering ports of discovered devices further comprises: transmitting an SMP Report General request from a first device to a neighboring device of the first device; and receiving an SMP Report General response in said first device from said neighboring device identifying the number of ports within said neighboring device.

Another aspect hereof further provides that the step of configuring further comprises: transmitting an SMP request from a first device to a second device wherein the SMP request includes vendor unique information identifying a routing attribute to be configured for a port of said second device.

Another aspect hereof further provides for recursively repeating the steps of the method to traverse devices of the SAS network domain to configure routing attributes of ports of devices of the SAS network domain.

Another feature hereof provides for an SAS network domain, comprising: a plurality of expander devices providing a plurality of ports within the domain wherein each port may have an associated routing attribute; and a domain control element coupled to at least one of the plurality of expander devices operable to configure routing attributes of the plurality of ports, wherein the domain control element is operable to configure the routing attributes of the ports by traversing port connections between the expander devices.

Another feature hereof provides an SAS network domain comprising: means for discovering the topology of the SAS network domain by traversing port connections between devices of the domain; and means for configuring SAS routing attributes associated with ports of devices of the domain in response to discovery of the topology of the domain.

Brief Description of the Drawings

Figure 1 is a block diagram of an SAS network domain in which features and aspects hereof have configured routing attributes.

Figure 2 is a flowchart illustrating a processing aspects hereof.

Figure 3 is a flowchart illustrating further details of automated routing attribute configuration features and aspects hereof.

Figure 4 is a block diagram of an SAS network domain in which ports of devices need be configured in accordance with features and aspects hereof.

Figure 5 depicts a chronological sequence of SMP messages exchanged in accord with features and aspects hereof to provide automated routing attribute configuration.

Figure 6 is a block diagram of the SAS network domain of figure 4 after automated routing attribute and routing table configuration has been performed in accord with features and aspects hereof.

Detailed Description of the Drawings

While the invention is susceptible to various modifications and alternative forms, a specific embodiment thereof has been shown by way of example in the drawings and will herein be described in detail. It should be understood, however, that it is not intended to limit the invention to the particular form disclosed; but rather, the invention is to cover all modifications, equivalents and alternatives falling within the spirit and scope of the invention as defined by the appended claims.

With reference now to the figures and in particular with reference to figure 1, an embodiment hereof is shown in SAS network domain 100. In this embodiment, SAS network domain 100 (i.e., through initiator device 101) can automatically configure ports to operate according to certain routing attributes. Such a configuration of ports may be performed prior to, or concurrently with, a discovery process as discussed above to discover the topology of the domain 100 and to determine routing tables for communicating between initiators and targets.

SAS network domain 100 includes SAS initiators 101, a plurality of expander devices 102 and a plurality of target devices 103. Ports of each device (101, 102 and 103) are shown with labels "D", "T" or "S" indicating completion of routing attribute configuration determined as "Direct routing attribute", "Table routing attribute" or "Subtractive routing attribute", respectively. Domain 100 illustrates an exemplary SAS network domain in which SAS initiators 101, such as SAS initiator I1, route data and/or control information through a network of expander devices 102 to various target devices 103. Exemplary of such target devices 103 are individual disk and tape drives as well as storage subsystems such as RAID storage systems and tape libraries. Accordingly, the SAS initiators 101 may direct read requests and/or write requests, to access data within target devices 103. In a related manner, data or status information generated by a target device 103 in response to such a request may be returned over the same or another path to the requesting initiator 101.

Expander devices 102 are configured for expanding a number of ports within SAS network domain 100. Each SAS device, such as initiators 101, expander devices 102 and target devices 103, in SAS network domain 100 is configured with a specific SAS address within the domain (also referred to herein as a device name, a GUID, or a World-Wide Name (WWN)). The number of SAS addresses within an SAS network

domain is governed by SAS standards. In general, SAS addresses or WWNs are intended to be globally unique identifiers (GUIDs) in accordance with the applicable SAS standards.

In routine operations of the SAS network domain 100, a SAS initiator 101 performs as a domain control element that determines the topology of the network domain. For example, the domain control element initiator 101 (I1) sends an SMP Discover request to an adjacent expander device 102 (E1) upon initialization of SAS network domain 100. The discovery process operable within the control element traverses the topology of SAS network domain 100, issuing Discover requests and other SMP commands to determine connections among the expander devices 102. Based upon retrieved information from the discovery process, the SAS initiator configures routing tables within the expander devices 102 (and within initiator device 101) such that data and/or control information may be routed between the SAS initiators 101 and the target devices the 103 via the network of expander devices 102.

As noted above, the discovery process as presently practiced presumes that all ports of all devices have been configured as to the type of routing attribute associated with each (i.e., direct, table or subtractive routing attribute values). As presently practiced, port configuration of routing attributes is manually performed.

In one feature hereof, SAS initiator 101 automatically configures routing attributes of all ports by similarly traversing port connections to determine connections between the ports and to thereby determine the topology of the SAS domain. The SAS initiator 101 may also thereby configure the ports of each expander connected to another expander device to operate according to appropriate routing characteristics (e.g., to operate as a subtractive routing port or as a table routing port based on a flow of traffic between the SAS initiator 101 and target devices 103). As noted, configuration of direct routing attribute ports is typically performed within each device as a part of the standard exchanges of frames between neighboring devices during reset operations. With the routing attributes of ports so determined by the automated features hereof, a discovery process as discussed above may then determine and configure routing tables to identify paths between initiators and targets through the various intermediate expander devices.

Either of SAS initiators I1 or I2 may perform the traversal algorithm. One or the other may be selected by any desired means as the preferred control element to perform the automated route attribute configuration. In another aspect hereof, multiple initiators I1 and I2 may begin the traversal processing to configure port attributes. Each initiator so processing the automated configuration process may discontinue the process (i.e., defer its processing) when it encounters another control element having a "higher priority." Such deferral avoids inconsistent results being configured by multiple configuration processes. Determining which of multiple initiators is "higher" or "lower" in priority and thus will defer its processing may be by any of several equivalent techniques. Since each initiator may discover the SAS address of the other, the higher SAS address initiator may defer to a lower SAS address initiator. Or, for example, a lower SAS address device may defer to a higher SAS address device. For example, SAS initiators I1 (lower SAS address) and I2 (higher SAS address) may both begin traversing SAS network domain 100. When I1 detects that I2 is also traversing SAS domain 100 (i.e., by discovering initiator I2 in its traversal of the SAS domain), I1 may defer to I2 to complete the traversal of SAS domain 100. The device that completes the configuration traversal process may therefore be referred to herein as "higher priority" or simply "preferred."

In traversing SAS network domain 100, the preferred SAS initiator may configure an SMP Report General request and/or a Discover request and transfer the request(s) to the expander device E1. In response to receiving the request(s), expander device E1 may respond with information about its ports. Among the information returned may be indications of the device name (i.e., WWN, SAS address or GUID) of the device attached to each port of expander E1. For example, a series of Discover requests directed to expander E1, accessing its ports in succession, will reveal that initiator I2 and expanders E2 and E3 are coupled to various of the ports of expander E1.

The control element (e.g., initiator I1) processes the response of expander device E1 and determines an appropriate routing configuration for each port (i.e., subtractive routing or table routing). In particular, in the example of figure 1, ports of E1 coupled to either E2 or E3 are configured as table routing ports as messages will generally be directed from initiators through these ports deeper into the SAS domain

network. The SAS initiator may configure those ports through a vendor specific message to expander device E1. Certain SMP commands of the SAS standards have vendor specific fields that a vendor may tailor to meet the needs of the vendor's systematic design. Examples of such commands and vendor specific fields are generally known to those killed in the art and are documented in SAS specifications.

The control element (e.g., initiator I1), having now "discovered" expander devices E2 and E3 are coupled to E1, may now issue similar SMP commands to each of expander devices E2 and E3. These commands traverse deeper into the SAS domain structure to allow the control element to continue configuring ports of expander devices. For example, as shown in figure 1, ports of E3 that are coupled to expander E1 may be configured by the control element for subtractive routing while the remaining ports of E3 are configured for table routing since the Discover response will indicate their coupling to expanders E4 and E5. In like manner, the control element may configure ports of E2, E4 and E5 to complete the process.

In one aspect hereof, the traversal of the SAS domain for purposes of configuring routing attributes may be performed as a separate process from the subsequent discovery process to determine and configure routing tables. However, in such an embodiment, some minimal routing information will be required to permit the complete traversal of the devices in the SAS domain for purposes of configuring routing attributes. For example, as expander devices are discovered and configured with appropriate routing attributes, devices already traversed in the SAS domain may have routing table entries configured to establish at least one path through each expander device to a next device deeper in the SAS domain hierarchy. In particular, when I1 (the control element) discovers that E4 and E5 are coupled to E3 by commands routed through E1 to E3, routing table entries may be configured within E1 indicating paths to E4 and E5 through E3. Such minimal routing table entries allow the continued traversal of expander devices in the SAS domain until all devices have been discovered and configured.

In another aspect hereof, the configuration of routing attributes and the generation and dissemination of complete routing table information may be integrated in a single process. Such a single process may traverse the SAS domain devices once but in that traversal may first configure routing attributes automatically and then

generate and disseminate routing table entries. The generation and dissemination of routing table entries in such a discovery process is well known in the art and is exemplified in the above noted appendix of the SAS specifications.

In still another aspect hereof, expander devices 102 themselves may traverse the SAS domain to configure routing attributes of ports of expanders within the domain. Upon initialization, each of expander devices 102 may begin traversing SAS domain 100 using the processes noted above. As noted above with respect to multiple initiators performing the automated configuration process, when an expander device encounters another expander device with a "higher priority", the lower priority expander device may defer to the higher priority expander device to complete the automated configuration processing. For example, when expander device E2 encounters expander device E1 in SAS domain 100, expander device E2 may determine that expander device E1 has a higher SAS address. Accordingly, E2 may cease traversal of SAS domain 100, leaving E1 to continue traversing and configuring routing attributes of ports of discovered expander devices. During this configuration process, expander devices may keep all end devices from addressing the domain by rejecting any requests from end devices to open a connection. At the conclusion of the configuration process, vendor-specific SMP messages may be used by the higher priority expander to notify all other expanders in the domain that configuration is complete and that requests by end devices may now be serviced.

Further details of processing features and aspects hereof to provide automated configuration of routing attributes in a SAS domain such as exemplified in figure 1 are provided herein below. Those skilled in the art will recognize that features and aspects hereof should not be limited to the particular numbers of SAS initiators, expander devices and/or target devices shown in figure 1. Rather, the traversal processes and structures as described herein for automated routing attribute configuration may be applied to numerous combinations of SAS devices and SAS network domain configurations.

Figure 2 is a flowchart providing a high level view of automated configuration features hereof. The method of figure 2 may be operable in response to a power on reset ("POR") of devices in a SAS domain or in response to detection of a change in the SAS domain topology by one of the SAS devices. As documented in SAS

specifications, any device that detects a change in the SAS domain configuration may generate and broadcast a message indicating detection of a change. In response thereto, one or more SAS devices in the domain may initiate processing to discover the new topology of the SAS domain and, in accordance with features and aspects hereof, to configure routing attributes of ports of the SAS domain.

In response to detection of such a change, element 200 is first operable to automatically configure routing attributes for ports or Phys of the various devices of the SAS domain. As discussed above, present practice presumes that such routing attribute configuration is performed as a manual process prior to a discovery process used to determine and configure routing tables in all devices of a SAS domain. Following the automatic configuration of routing attributes by element 200, element 202 is operable to automatically configure routing tables in all devices of the SAS domain. As noted above, an exemplary discovery process for determining routing information for all devices of a SAS domain is provided as an appendix in SAS specifications and is therefore well known to those of ordinary skill in the art.

Element 204 continues or initially commences normal operation of the SAS domain following completion of the automatic configuration of routing attributes and the automatic discovery process to determine and configure routing tables for all devices of the SAS domain. As part of the processing of element 204, configuration information for routing attributes of each port or Phy of each device may be written to non-volatile configuration memory of each device. In addition, routing table entries generated by the discovery process may be written to the devices of the SAS domain. This configuration information permits the commencement or continuation of normal operations of the SAS domain devices to forward SCSI commands through the serial attached network of the SAS domain.

The dashed line 250 surrounding elements 200 and 202 represents an optional feature hereof wherein the automatic configuration of routing attributes may be combined with the automated discovery process for configuring routing tables such that both processes are performed as a single, unified operation traversing the devices of the SAS domain once. In addition, element 204 as noted above is operable to configure information determined by both processes (200 and 202) into associated configuration memories of each device of the SAS domain. Where the two processes

are not tightly integrated as a single process, the configuration information so determined may be written as two distinct steps – routing attribute information may be written to the SAS devices as part of the processing of element 200 while routing table information may be written to the SAS device as part of the processing of element 202 or 204. Details of such design choices are discussed further herein below.

Figure 3 is a flowchart providing additional details of exemplary embodiments of features and aspects hereof for automatic configuration of routing attributes associated with ports or Phys of devices in a SAS domain. Numerous optional design choices and alternatives are depicted within the elements of figure 3 as discussed further herein below.

In the exemplary embodiments presented in figure 3, the process is implemented as a recursive algorithm performing a depth-first traversal of the devices in the SAS domain. The depth-first traversal commences from a predetermined top-most device of the SAS domain hierarchy. Routing attributes are, in general, determined and configured relative to an identified SAS initiator device in the SAS domain. However, as discussed further herein below, the configuration algorithm may operate within any device of the SAS domain so long as the routing attribute configuration is properly determined as routing attributes relative to some SAS initiator of the domain. The top-most initiator may be determined, as noted above, based on a priority determination. A highest priority initiator device of the SAS domain may be identified by, for example, the highest SAS address (i.e., WWN, device name or GUID) among all initiator devices, or, for example, as the lowest SAS address among all the initiator devices.

As noted above with respect to figure 2, the routing attribute configuration process of figure 3 may be operable in response to POR or other reset of devices of the SAS domain or may be operable in response to a detected change in the topology of the SAS domain.

In general, the method of figure 3 is operable to process each neighbor device of the SAS device performing the process. For each neighboring expander device, an SMP Report General request is formulated and transmitted to determine the number of ports or Phys associated with that device. In addition, an SMP Discover request

returns additional information for a device indicating the device name of each device attached to a port or Phy of the responding device. For each port or Phy of a neighboring expander device, the process is recursively invoked to look for neighbors of that neighboring device and so on. As devices are discovered along the traversal of the SAS domain and the type of the device is determined, the routing attributes for each port may be determined and configured. This automated approach eliminates the errors and delay inherent in manual processes presently practiced.

Element 300 is first operable with respect to a present device from which the process is operating (referred to in figure 3 as "This Device") to determine whether additional neighboring devices remain to be processed. If all neighboring devices of This Device have been processed, elements 302 through 306 are operable to perform a return for the present invocation of the recursive process. As noted above, in the depicted exemplary embodiment, a recursive algorithm may be used to provide a depth-first traversal of all devices in the SAS domain. Elements 302 through 306 therefore provide for a return from the present level of recursive invocation to a previous level of recursive invocation. The topmost level of invocation therefore completes the recursive processing and completes the process of figure 3. Such recursive programming paradigms are well known to those of ordinary skill in the art. In particular, as shown in figure 3, element 302 determines whether the present invocation of the process is the topmost initiating invocation of the recursive algorithm. If not, element 304 performs a recursive return from the present invocation level to an previous recursive invocation level to complete processing of a previous device. Processing then continues by looping back to element 300 to continue processing at an earlier recursion level. If element 302 determines that the present recursive level is the highest level of the recursive hierarchy, element 306 is then operable to signify completion of the automated routing attribute configuration operation. Element 306 permits commencement or continuation of normal operation of devices in the SAS domain.

If element 300 determines that additional neighboring devices of This Device remain to be processed, element 308 is operable to formulate and transmit an SMP Report General request to the "Next Neighboring Device" of This Device. As known in the art and as documented in SAS specifications, a Report General request returns,

among other items of information, the number of ports or Phys associated with the device processing and responding to the Report General request. Further, as is known in the art, each device of a SAS domain is aware of its direct neighboring devices. As part of the Phy Reset Sequence (described in the SAS specifications), Identify Frames are exchanged between neighboring devices whose Phys are directly coupled. With such information, element 308 may transmit a request for information to each of the direct neighbors of This Device without requiring routing table entries to determine a path to forward such a message. However, as requests descend deeper into the hierarchy of the SAS domain, at least some minimal routing information is useful as discussed further herein below.

The process next iterates through each of the ports or Phys of the responding Next Neighbor Device. In particular, element 310 determines if all Phys of the present Next Neighbor Device have been processed. If so, element 312 is operable to prepare for processing of another Next Neighbor Device of This Device. Processing then continues by looping back to element 300 for processing of remaining neighbors, if any, of This Device.

If element 310 determines that additional ports or Phys of the present Next Neighbor Device remain to be processed, element 316 is operable to formulate and transmit an SMP Discovery request to the present port or Phy of the present Next Neighbor Device. As is known in the art and as documented in the SAS specifications, a Discovery request retrieves information from the identified device. Among the information returned from such a request is the globally unique identifier (i.e., WWN, GUID, device name or SAS address) of the device coupled to a port or Phy of the identified Next Neighbor Device.

As noted above, multiple devices in the SAS domain may initiate the processing depicted in this figure 3. In such an embodiment, a first device of the multiple devices may complete the automated configuration process while other devices should defer to the first device. As further noted above, such a determination of which device may complete the process may be based on any of numerous prioritization factors. For example, a higher SAS address may be deemed to be a higher priority as compared to another device with the lower SAS address. Conversely, a lower SAS address could be deemed to be a higher priority than a

higher SAS address. Those of ordinary skill in the art will readily recognize that any prioritization scheme may be utilized to determine which of multiple SAS devices should be permitted to complete processing of the configuration process. Element 318 is therefore operable to determine from the returned information in response to a Discovery request whether a newly discovered device coupled to a port or Phy of the Next Neighbor Device indicates that it has a higher priority than the device in which the process is presently operating. If so, the process completes within the present device by deferring to the higher priority device to complete the routing attribute configuration process. The dashed lines of element 318 indicate that it represents an optional design choice element that may be excluded where other techniques are utilized to determine which SAS device will perform the automated configuration process of this figure 3. Numerous other approaches for selecting a device to complete the processing will be readily apparent to those of ordinary skill in the art.

Processing may continue in the higher priority device at element 320 by propagating upstream from This Device routing table information useful for configuring routing tables entries in hierarchically higher devices. As used herein, "upstream" refers to transmitting routing table information to expander devices previously processed in the recursive processing of this figure 3.

As noted above, generation and transmission of routing table entries in the SAS domain may be integrated with the automated configuration features and aspects hereof for configuring the expander Phys' routing attributes. Alternatively, the routing attribute configuration processes hereof may be performed as a separate preliminary step before discovery processing is used as exemplified in the SAS specifications to generate and propagate routing table entries. The dashed lines surrounding element 320 indicate that it is an optional design choice where the configuration of routing attributes is to be closely integrated with generation and propagation of routing table information. Where automated routing attribute configuration is performed as a preliminary step independent of all routing table generation and propagation, processing of element 320 may be skipped.

Element 322 is then operable to determine whether the device attached to the responding port or Phy of the Next Neighbor Device indicates that it is an end device type. An end device type, as known in the art and as provided in the SAS

specification, is either an initiator device or a target device – i.e., start and end points for SCSI command exchanges through the SAS domain. If element 322 determines that the newly discovered device is an end device type, no further processing is necessary to configure the routing attribute of the port and processing continues by looping back to element 310 to process additional ports or Phys of the present Next Neighbor Device. Otherwise, element 324 determines whether the newly discovered device is an expander device type (either an expander or a fan-out expander). If not (i.e., the port is not in use), no further processing is necessary for this port or Phy of the present Next Neighbor Device and processing continues by looping back to element 310 to process any additional ports or Phys of the present Next Neighbor Device.

If element 324 determines that the newly discovered device is an expander device, element 326 is operable to configure routing attributes of the Phy of the Next Neighbor Device and that of the newly discovered expander port. For example, the present port of the Next Neighbor Device is configured for Table Routing attribute while the corresponding port of the newly discovered expander device is configured for Subtractive Routing attribute (or for Table Routing where the newly discovered expander is a fan-out expander device). SMP commands containing vendor unique fields may be used for this purpose as noted above. Such commands and their respective fields are known to those of ordinary skill in the art and are documented in the SAS specifications.

As noted above, inclusion of optional element 320 allows routing attribute configuration processing to be integrated with the known discovery process used for generating and propagating routing table entry information. Even if the routing attribute configuration process of figure 3 is not so integrated with generation and propagation of routing table entries, some minimal routing information is useful to continue the recursive, depth-first traversal from the device performing the process of figure 3 through the newly discovered expander device. In that case, element 328 is next operable to generate and propagate upstream minimum routing table information required to permit the recursive process to continue the depth-first, hierarchical traversal of devices in the SAS domain. Lastly, having discovered yet another expander device in the hierarchical SAS domain, element 330 is next operable to

recursively invoke the same process but using the present Next Neighbor Device as This Device and using the newly discovered expander device as a Next Neighbor Device. The process then continues with element 330 by invoking the same recursive process until all SAS devices of the SAS domain have been discovered and have had their respective routing attribute values set for each port or Phy associated there with.

As new General Report requests and Discover requests are generated and transmitted, the minimal routing information generated by element 328 or the full routing information generated by element 320 is may be relied upon to assure that the SMP messages are forwarded through the SAS domain to the next device to be discovered and configured.

The exemplary processes discussed above may also be understood with the aid of a configuration example. Figure 4 depicts a simple, exemplary SAS domain including a single initiator ("I1"), four target devices ("T1", "T2", "T3" and "T4") and three expander devices coupling the target devices to the initiator ("E1", "E2" and "E3"). As depicted, initiator I1 has two ports or Phys – ports P1 and P2. Targets T1 and T2 each have a single port while target T3 has two ports. Expanders E2 and E3 each have three ports while expander E1 has two ports.

As shown in figure 4, initiator device I1 and target devices T1, T2 and T3 each configure their respective ports with routing attributes as "Direct" or as end device types. In like manner, each expander (E1, E2 and E3) configures the port attribute as "Direct" (or "D") for any port directly coupled to an initiator device or a target device. This minimal information for Direct routing attribute configuration is readily available to each SAS device by virtue of the standardized information transmitted in Identify frames exchanged as a part of standard power on or other reset processing. Each device in a SAS domain exchanges such Identify frames with its direct neighbors to determine what type of device is coupled as an neighbor on each of its ports. This Identify frame information is sufficient for each device to exchange further messages with any of its direct neighbors and represents, in effect, routing information to transfer messages to any of its direct neighbors, or to forward messages from a first neighbor to a second neighbor. Details regarding structure and content of Identify frame messages are provided in the SAS specifications.

As can be seen in figure 4, port P3 of expander E2 and port P1 of expander E3 do not have a routing attributes associated with therewith. Whereas prior processes would require manual procedures to configure the routing attributes for these ports, features and aspects hereof allow for automated configuration of these and other routing attributes.

As noted above, automated routing attribute configuration features hereof may operate within initiator devices of the SAS domain or may operate within expander devices of the SAS domain. Further, as noted above, the processing may commence in multiple devices (i.e., multiple initiators or multiple expanders) but only one of the devices will complete the process. Other devices will defer to a first, higher priority, device operating processes hereof.

Figure 5 represents an exemplary exchange of messages to permit automated configuration of routing attributes in expander devices of the exemplary SAS domain of figure 4. These exemplary message exchanges of figure 5 presume that the process for automated configuration is operable within initiator I1 as shown in figure 4. Those of ordinary skill in the art will recognize that the message exchange depicted utilizes SMP messages to configure routing attributes in the above identified ports lacking such configuration. As a matter of design choice, other messages may be exchanged within the SAS domain to perform the process for such automated routing attribute configuration. For example, all messages may be vendor defined messages representing out-of-band messages for the SAS domain media and protocols.

Figure 5 depicts a chronological sequence of messages exchanged between SAS devices I1, E2 and E3 as shown in figure 4. Relative time of the chronological sequence of messages progresses in the downward direction of the figure. The sequence depicted represents a relevant portion of an exemplary message exchange involved in features and aspects hereof to permit automated routing attribute configuration. In figure 5, three devices (I1, E2 and E3) are shown as column headers near the top of the figure. As time progresses downward from top to bottom in the figure, each message is indicated by a directed arrow from a first device transmitting the message to a second device receiving the message. Descriptive information regarding a message appears underneath each corresponding arrow providing some detail of the message content and meaning.

As shown in figure 5, a port ("P*n*") of a device (i.e., I*m*, E*m*, or T*m*) may be referred to herein using the notation "I*m*.P*n*", "E*m*.P*n*" or "T*m*.P*n*" (where *m* and *n* are device and port numbers, respectively). For example, I1.P1 refers to port P1 of initiator device I1. Further, the Device Name (i.e., WWN, GUID or SAS address) for a device is indicated by the same identifier shown in the figure but in quotes. For example, the WWN for device E1 is denoted herein as "E1."

Referring again to figure 5, message 500 is an SMP Report General request directed from initiator I1 to expander E2. Message 502 is the response generated by expander E2. As specified in the SAS specifications, the response to the Report General request provides the total number of ports or Phys presently in use by the responding device (here 3 ports on expander device E2).xx

Next, three SMP Discover Request messages 504 are directed from initiator I1 to expander E2. Three such Discover requests are forwarded from I1 to E2 in response to the Report General Response discussed above indicating three Phys on expander E2. Message 506 represents the Discover Responses generated thereto and directed from expander E2 back to initiator I1. Each Discover Response includes the Device Name of a corresponding device attached to each port of the responding device – here expander E2. In particular, in the example of figure 4, expander E2 reports in a first response that on its port P1 is attached device "I1"; reports in a second response that on its port P2 is attached device "E3" and reports in a third response that on its port P3 is attached device "T1". Those of ordinary skill in the art will recognize that Discover Requests and corresponding Discover Responses are typically sent and received, respectively, in alternating sequence. The multiple requests and responses are shown as a sequence of requests followed by a sequence of responses for simplicity of the figure.

Having received these Discovery Responses from expander E2, initiator I1 may configure its internal target list indicating that devices T1 and E3 are both accessible through expander E2. Such a configuration message is indicated by curved arrow 508 the curved nature of which indicates processing within initiator I1 (i.e., no message need be sent between I1 and any other device). As noted above, configuration of routing table entries (or internal target lists) may be minimally performed to permit continued operation of the routing attribute configuration features

hereof or may be complete routing table configuration where the processes hereof are integrated with an existing discovery process for generation and dissemination of routing information.

In view of the Discover response received at message 506, initiator I1 may transmit message 510 to expander device E2 requesting that it configure its port P3 with a table routing attribute. As noted above, such a message may be an SMP message with vendor unique information requesting configuration of such a routing attribute. The command may include directions for the receiving device to program its non-volatile memory to configure the indicated routing attribute or the message may itself implement the required re-programming of programmable memory within the receiving device.

Initiator I1 is now aware of another expander device in the SAS domain, namely, expander E3 coupled to expander E2. Processing features hereof may therefore recursively invoke the same process to descend deeper in the traversal of the SAS domain structure. In particular, the recursive invocation may exchange similar messages with expander E3 to discover more of the topology of the SAS domain. Following such a recursive invocation, message 512 may be generated and represents a Report General request transmitted from initiator I1, routed through expander E2, and addressed to expander E3. In response thereto, expander E3 generates response message 514 indicating that expander E3 has three ports or Phys. Such a response from E3 is returned via expander E2 to initiator I1. Initiator I1 next generates three Discover Request messages 516 addressed to expander device E3 and transmitted through expander E2. Three Discover Requests are generated in response to the Report General Response indicating three Phys on the expander E3. Expander device E3 then generates Corresponding three Discover Responses in message 518 indicating the device name of every device attached to its presently active ports. In particular, expander E3 indicates in a first response message that its port P1 is attached to expander device "E2", indicates in a second response message that its port P2 is attach to target device "T2" and indicates in a third response message that its port P3 is attached to target device "T3". As above, those of ordinary skill in the art will recognize that Discover Requests and corresponding Discover Responses are typically sent and received, respectively, in alternating sequence. The multiple requests and

responses are shown as a sequence of requests followed by a sequence of responses for simplicity of the figure.

As above, having received such information, initiator I1 may add entries in its internal target list indicating that target devices T2 and T3 are known to be accessible via expander E2 (and thence via expander E3). Such a configuration process is indicated as message 520 and a curved arrow representing processing within initiator I1 to configure its own routing table entries or internal target lists. In addition, initiator I1 transmits message 522 to expander device E2 providing routing table entries to be added within expander E2 indicating that target devices T2 and T3 are both accessible via expander E3. Also as above, the routing information may be minimal information to permit completion of the routing attribute configuration features hereof or may be complete routing information where processes hereof are integrated with known discovery processing. Lastly, initiator I1 generates message 524 addressed to expander E3 indicating that its port P1 is to be configured with the subtractive routing attribute. Such a determination is now possible for initiator I1 in view of its understanding that expander E2 is coupled to expander E3 to thereby permit forwarding of messages to further target devices T2 and T3.

Recursive operations of initiator I1 may then return to complete other processing in the topology discovery and attribute configuration of SAS domain 400 of figure 4. Upon completion of such processing, figure 6 shows exemplary SAS domain 400 with routing attribute configuration completed in all expander ports -- namely port P3 of expander E2 and port P1 of expander E3. Specifically, E2.P3 is configured for table routing attribute while E3.P1 is configured for subtractive routing attribute.

In addition as shown in figure 6, routing information in routing table entries associated with initiator I1 and expander E2 may be completed by the processing described above. As discussed above, the generation and propagation of routing table entries and the automated routing attribute configuration features and aspects hereof may be integrated as a single process or may be implemented as distinct processes. Therefore, routing entries generated and propagated as described herein with respect to figure 5 may be complete routing information where the routing attribute features hereof are tightly integrated with the SAS discovery process described in SAS

specification. Alternatively, the routing table entries described herein with respect to figure 5 may be minimal routing table entries to permit completion of the routing attribute configuration as a step preceding the complete routing table generation of the SAS discovery process.

Those of ordinary skill in the art will recognize that figures 4 through 6 are merely intended as exemplary of a simple SAS domain and processing features and aspects hereof to provide automated routing attribute configuration. Those of ordinary skill in the art will further recognize that the exemplary processing described herein may be readily expanded to more complex SAS domains with any number of expander devices in the network. In addition, those of ordinary skill in the art will recognize both recursive and non-recursive programming techniques that may be utilized to implement features and aspects hereof. Still further, those of ordinary skill in the art will recognize that the processing features and aspects hereof may be equivalently implemented as suitably programmed instructions in a general or special purpose processor or may be implemented as electronic circuits specifically designed for providing at least the processing features described herein.

While the invention has been illustrated and described in the drawings and foregoing description, such illustration and description is to be considered as exemplary and not restrictive in character. One embodiment of the invention and minor variants thereof have been shown and described. Protection is desired for all changes and modifications that come within the spirit of the invention. Those skilled in the art will appreciate variations of the above-described embodiments that fall within the scope of the invention. As a result, the invention is not limited to the specific examples and illustrations discussed above, but only by the following claims and their equivalents.

Claims

What is claimed is:

1. An automated method of configuring routing attributes of ports within an SAS network domain, comprising:

automatically discovering devices of the SAS network domain;

automatically discovering ports of the discovered devices; and

automatically configuring routing attributes of discovered ports.

2. The method of claim 1 wherein the steps of discovering devices, discovering ports and configuring ports each include a step of exchanging SMP messages.

3. The method of claim 2 further comprising:

configuring routing table information within devices of the SAS network domain wherein said routing table information is sufficient to identify paths in the SAS network domain to enable the exchange of said SMP messages.

4. The method of claim 2 further comprising:

completely configuring routing table information to identify all paths for exchange of messages within the SAS network domain.

5. The method of claim 4 wherein the step of completely configuring is substantially integrated with the steps of discovering devices, discovering ports and configuring ports .

6. The method of claim 1 wherein the step of discovering devices further comprises:

transmitting an SMP Discover request from a first device to a neighboring device of the first device; and

receiving an SMP Discover response in said first device from said neighboring device identifying the other devices coupled to ports of said neighboring device.

7. The method of claim 1 wherein the step of discovering ports of discovered devices further comprises:

transmitting an SMP Report General request from a first device to a neighboring device of the first device; and

receiving an SMP Report General response in said first device from said neighboring device identifying the number of ports within said neighboring device.

8. The method of claim 1 wherein the step of configuring further comprises:

transmitting an SMP request from a first device to a second device wherein the SMP request includes vendor unique information identifying a routing attribute to be configured for a port of said second device.

9. The method of claim 1 further comprising:

recursively repeating the steps of the method to traverse devices of the SAS network domain to configure routing attributes of ports of devices of the SAS network domain.

10. An SAS network domain, comprising:

a plurality of expander devices providing a plurality of ports within the domain wherein each port may have an associated routing attribute; and

a domain control element coupled to at least one of the plurality of expander devices operable to configure routing attributes of the plurality of ports, wherein the domain control element is operable to configure the routing attributes of the ports by traversing port connections between the expander devices.

11. The SAS network domain of claim 10 wherein the domain control element comprises:

an SAS initiator device coupled to at least one of the plurality of expander devices.

12. The SAS network domain of claim 10 wherein the domain control element comprises:

an SAS expander device coupled to at least one of the plurality of expander devices.

13. The SAS network domain of claim 10 wherein the each of the expander devices comprises a routing table and wherein the domain control element is further adaptable to configure the routing tables of the expander devices.

14. An SAS network domain comprising:

means for discovering the topology of the SAS network domain by traversing port connections between devices of the domain; and

means for configuring SAS routing attributes associated with ports of devices of the domain in response to discovery of the topology of the domain.

15. The SAS network domain of claim 14 further comprising:

means for configuring routing tables in devices of the domain.

16. The SAS network domain of claim 15 wherein said means for configuring routing tables and said means for discovering and said means for configuring SAS routing attributes are substantially integrated so as to traverse port connection between devices of the domain only once.

17. The SAS network domain of claim 14 wherein the means for discovering the topology further comprises:

means for exchanging SMP messages between devices of the domain to identify devices and to identify ports of the devices and to identify connections between the ports of the devices.

18. The SAS network domain of claim 17 wherein the means for exchanging SMP messages further comprises:

means for exchanging an SMP Report General request and response messages to identify ports of devices and connections between ports of devices.

19. The SAS network domain of claim 17 wherein the means for exchanging SMP messages further comprises:

means for exchanging SMP Discover request and response messages to devices of the domain.

20. The SAS network domain of claim 14 wherein the means for configuring further comprises:

means for transmitting an SMP message having vendor unique information from a first device to a second devices to instruct the second device to configure the routing attribute of a port of the second device.

Abstract

Systems and methods are provided for automatically configuring ports of devices within an SAS network domain. A domain control element, such as an SAS initiator, is coupled to a plurality of expander devices. The domain control element configures ports of the expander devices by traversing port connections between the expander devices to determine routing attributes of the ports. The domain control element automatically configures the ports to operate according to the routing attributes. In one aspect hereof, an initiator device of the SAS network domain serves as a control element to perform the automated configuration of routing attributes. In another aspect hereof, an expander device serves as a control element to configure routing attributes of the ports.

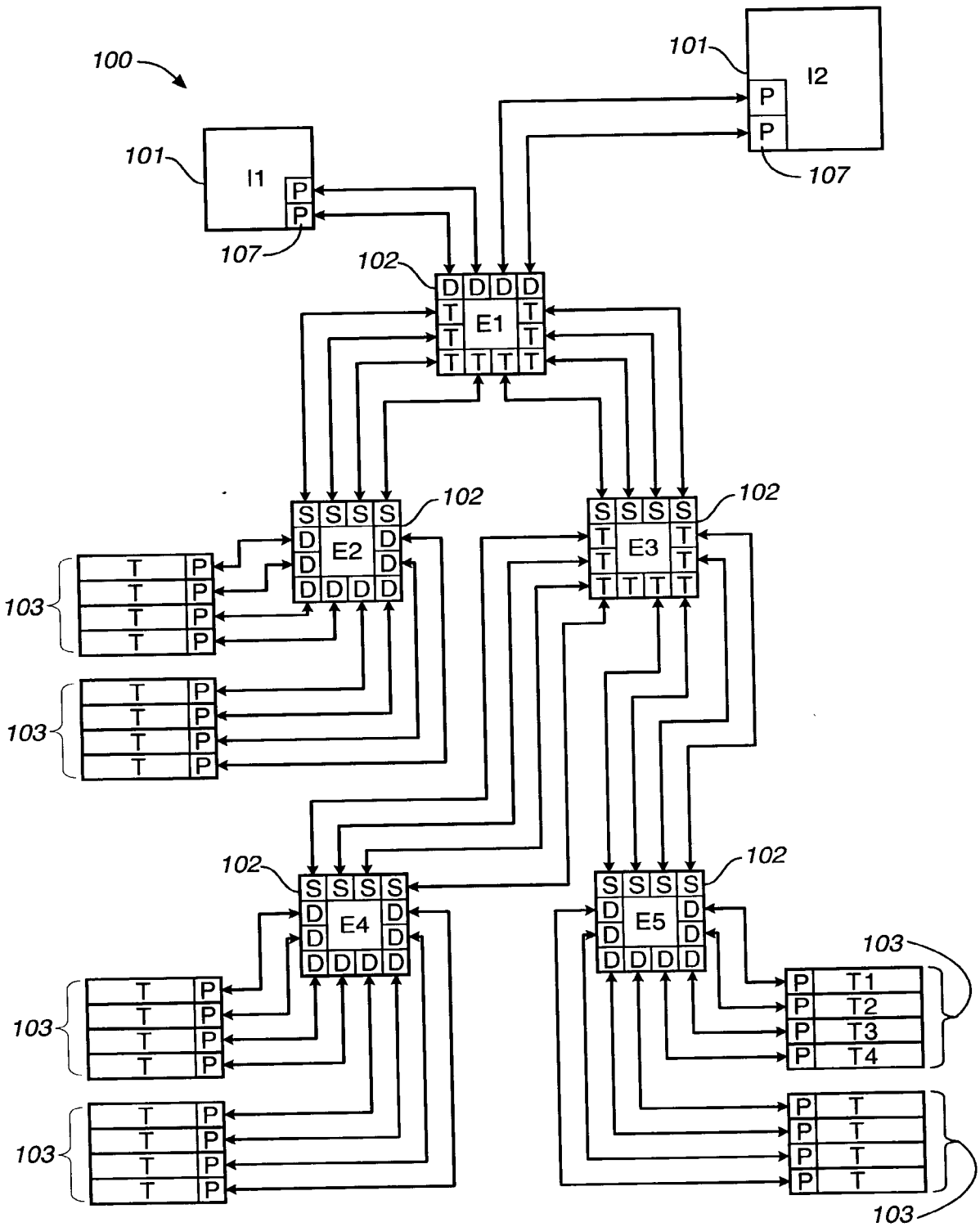

100
101
I1
P
P
107
101
I2
P
P
107
102
D D D D
T E1 T
T T
T T T T
102
S S S S
D E2 D
D D
D D D D
102
S S S S
T E3 T
T T
T T T T
103
T P
T P
T P
T P
103
T P
T P
T P
T P
102
S S S S
D E4 D
D D
D D D D
102
S S S S
D E5 D
D D
D D D D
103
T P
T P
T P
T P
103
T P
T P
T P
T P
103
P T1
P T2
P T3
P T4
P T
P T
P T
P T
103

FIG._1

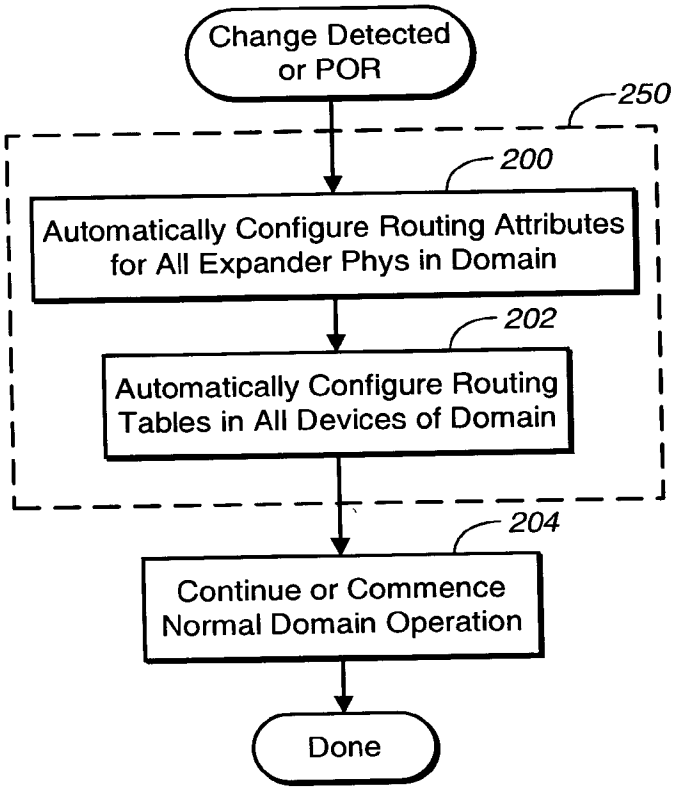
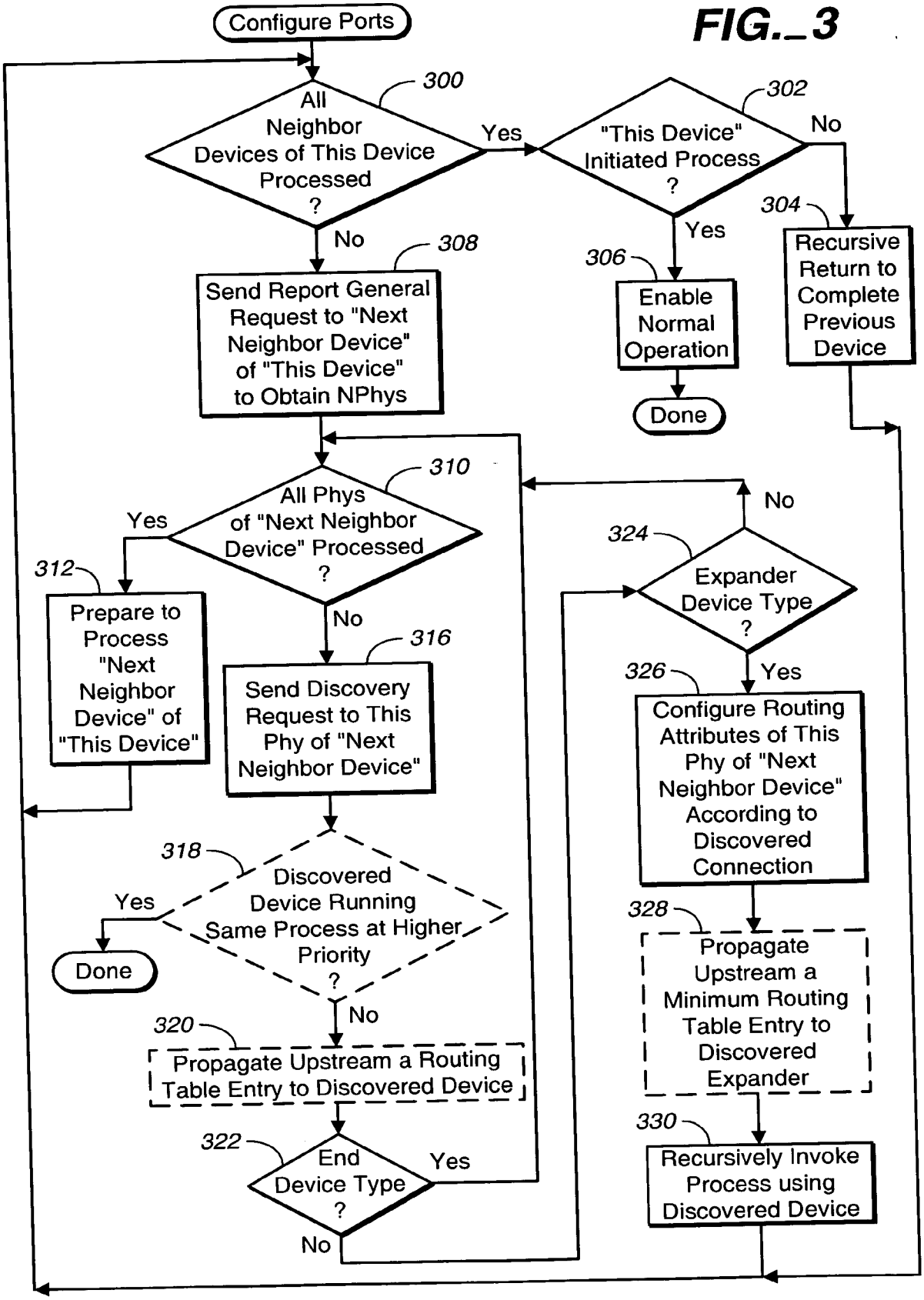

Change Detected or POR
250
200
Automatically Configure Routing Attributes for All Expander Phys in Domain
202
Automatically Configure Routing Tables in All Devices of Domain
204
Continue or Commence Normal Domain Operation
Done

FIG._2

FIG._3


Configure Ports
300
All Neighbor Devices of This Device Processed ?
Yes
302
"This Device" Initiated Process ?
No
304
Recursive Return to Complete Previous Device
Yes
306
Enable Normal Operation
Done
No
308
Send Report General Request to "Next Neighbor Device" of "This Device" to Obtain NPhys
310
All Phys of "Next Neighbor Device" Processed ?
Yes
312
Prepare to Process "Next Neighbor Device" of "This Device"
No
316
Send Discovery Request to This Phy of "Next Neighbor Device"
318
Discovered Device Running Same Process at Higher Priority ?
Yes
Done
No
320
Propagate Upstream a Routing Table Entry to Discovered Device
322
End Device Type ?
Yes
No
No
324
Expander Device Type ?
Yes
326
Configure Routing Attributes of This Phy of "Next Neighbor Device" According to Discovered Connection
328
Propagate Upstream a Minimum Routing Table Entry to Discovered Expander
330
Recursively Invoke Process using Discovered Device

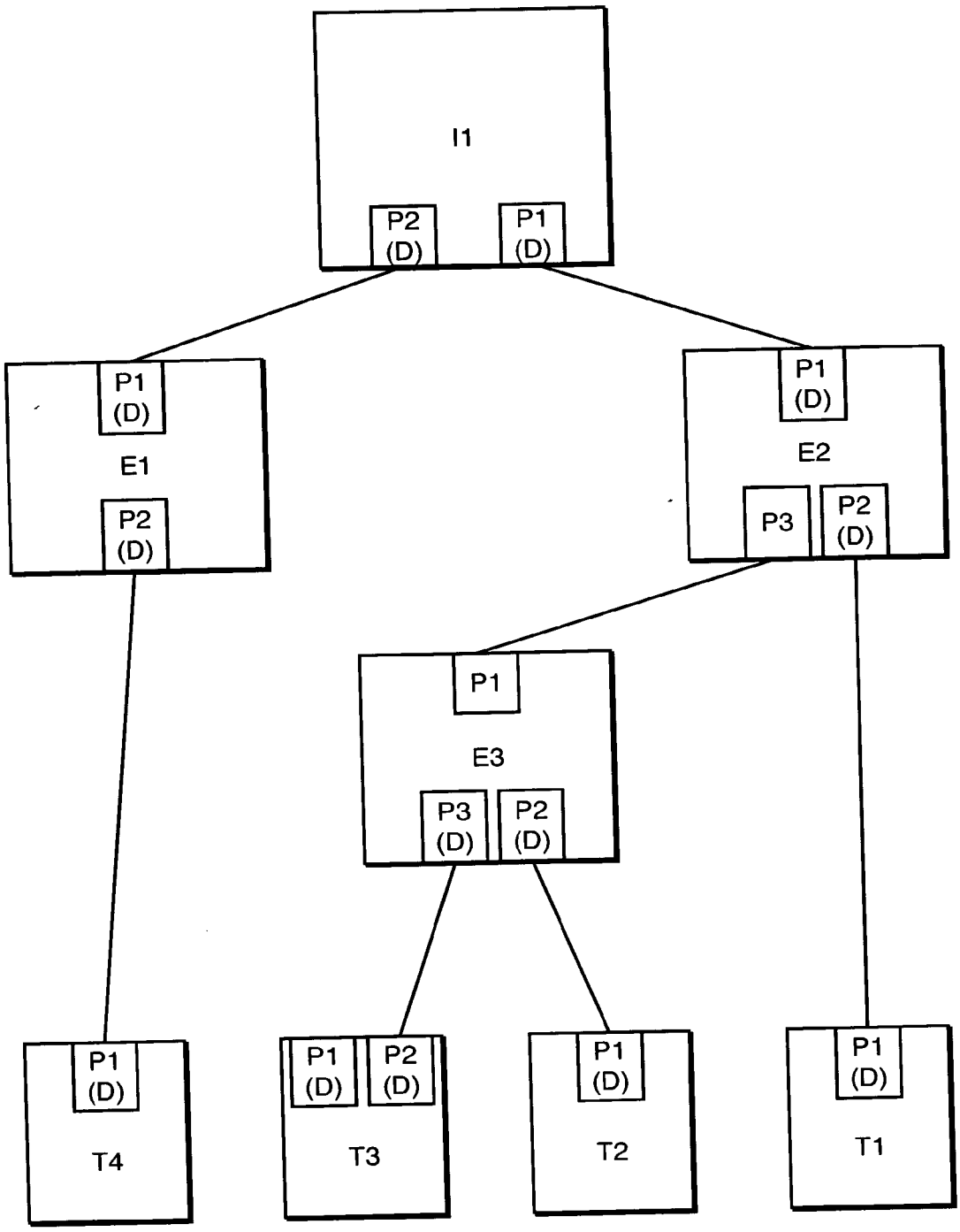

I1
P2 (D)
P1 (D)
P1 (D)
E1
P2 (D)
P1 (D)
E2
P3
P2 (D)
P1
E3
P3 (D)
P2 (D)
P1 (D)
T4
P1 (D)
P2 (D)
T3
P1 (D)
T2
P1 (D)
T1

FIG._4

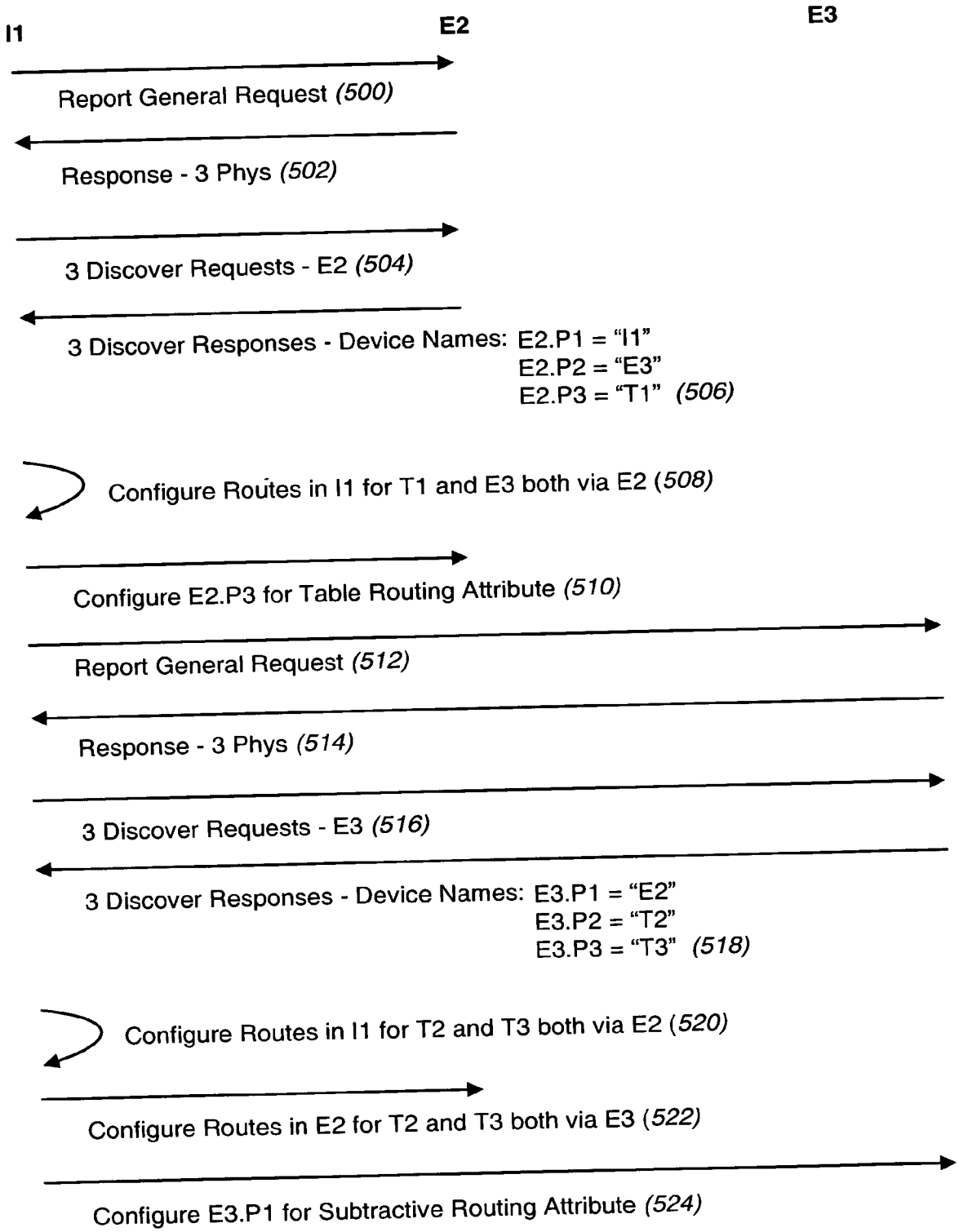

I1
E2
E3
Report General Request (500)
Response - 3 Phys (502)
3 Discover Requests - E2 (504)
3 Discover Responses - Device Names: E2.P1 = "I1"
E2.P2 = "E3"
E2.P3 = "T1" (506)
Configure Routes in I1 for T1 and E3 both via E2 (508)
Configure E2.P3 for Table Routing Attribute (510)
Report General Request (512)
Response - 3 Phys (514)
3 Discover Requests - E3 (516)
3 Discover Responses - Device Names: E3.P1 = "E2"
E3.P2 = "T2"
E3.P3 = "T3" (518)
Configure Routes in I1 for T2 and T3 both via E2 (520)
Configure Routes in E2 for T2 and T3 both via E3 (522)
Configure E3.P1 for Subtractive Routing Attribute (524)

FIG._5

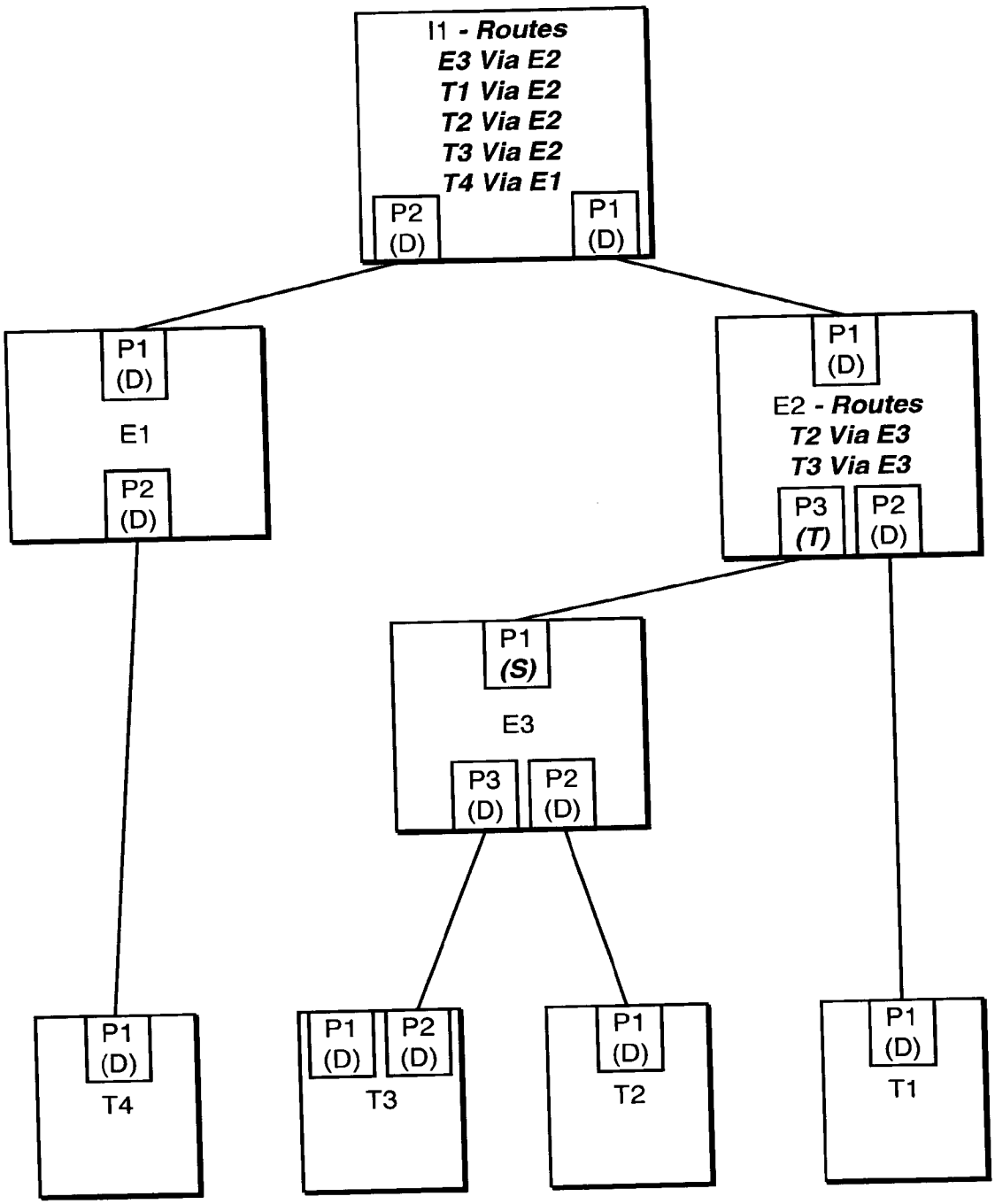

I1 - Routes
E3 Via E2
T1 Via E2
T2 Via E2
T3 Via E2
T4 Via E1
P2 (D)
P1 (D)
P1 (D)
E1
P2 (D)
P1 (D)
E2 - Routes
T2 Via E3
T3 Via E3
P3 (T)
P2 (D)
P1 (S)
E3
P3 (D)
P2 (D)
P1 (D)
T4
P1 (D)
P2 (D)
T3
P1 (D)
T2
P1 (D)
T1

FIG._6

US006199137B1

(12) **United States Patent**
Aguilar et al.

(10) **Patent No.: US 6,199,137 B1**
(45) **Date of Patent: Mar. 6, 2001**

(54) **METHOD AND DEVICE FOR CONTROLLING DATA FLOW THROUGH AN IO CONTROLLER**

(75) Inventors: **Raul A. Aguilar**, Hellertown; **Kevin Joseph Lynch**, Slatington; **James Thomas Clee**, Orefield; **James Edward Guziak**, Laurys Station; **Farrukh Amjad Latif**, Lansdale, all of PA (US)

(73) Assignee: **Lucent Technolgies, Inc.**, Murray Hill, NJ (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/477,591**

(22) Filed: **Jan. 4, 2000**

Related U.S. Application Data

(60) Provisional application No. 60/114,771, filed on Jan. 5, 1999, provisional application No. 60/114,772, filed on Jan. 5, 1999, and provisional application No. 60/114,767, filed on Jan. 6, 1999.

(51) **Int. Cl.**[7] **G06F 13/40**

(52) **U.S. Cl.** **710/129**; 710/29; 710/37; 710/10

(58) **Field of Search** 710/8, 10, 11, 710/16, 36, 29, 37, 42, 101–103, 105, 128, 129

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,671,441	*	9/1997	Glassen et al.	710/8
5,815,731	*	9/1998	Doyle et al.	710/10
5,892,928	*	4/1999	Wallach et al.	710/103
5,935,228	*	8/1999	Shinomura	710/102
5,964,852	*	10/1999	Overton	710/62
6,003,097	*	12/1999	Richman et al.	710/8
6,058,445	*	5/2000	Chari et al.	710/103

* cited by examiner

Primary Examiner—Ayaz R. Sheikh
Assistant Examiner—Raymond N Phan
(74) *Attorney, Agent, or Firm*—Schnader Harrison Segal & Lewis LLP

(57) **ABSTRACT**

An IO controller device and method for controlling data flow, the method including determining a desired configuration for the IO controller, reprogramming the IO controller to allow for processing of one or more descriptor lists, modifying the configuration of the IO controller to reflect the addition or deletion of one or more virtual controllers, re-enumerating the IO controller, and processing a descriptor list for each of the IO controller and the one or more virtual controllers. The integrated circuit device for use as an IO controller includes a system bus interface, a programmable list processor and a port router. The integrated circuit device is adapted for reconfiguration to add or delete one or more virtual controllers. The virtual controllers provide substantially the full bandwidth supported by the integrated circuit device. The IO controller device and apparatus may be applied to personal computer systems, information appliances, set-top boxes, cable modems, game consoles, smart appliances, handheld computers, palm-sized computers, embedded control systems, workstations, servers and the like.

12 Claims, 4 Drawing Sheets

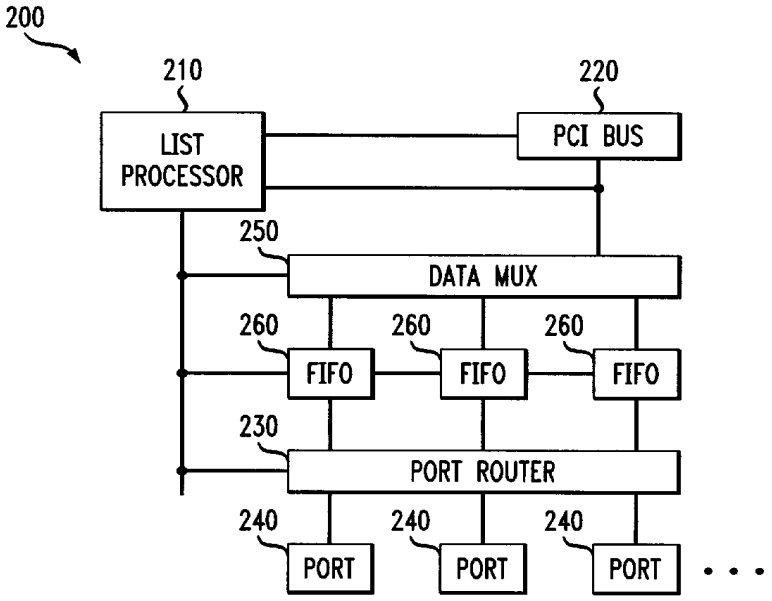


FIG. 1

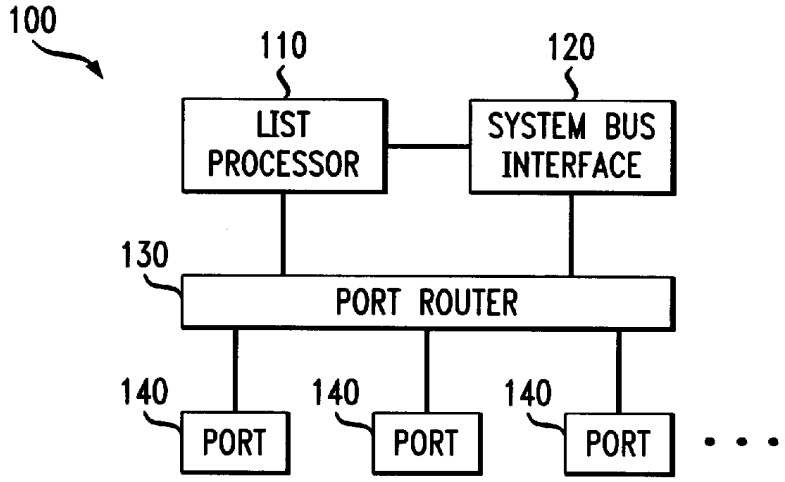


FIG. 2

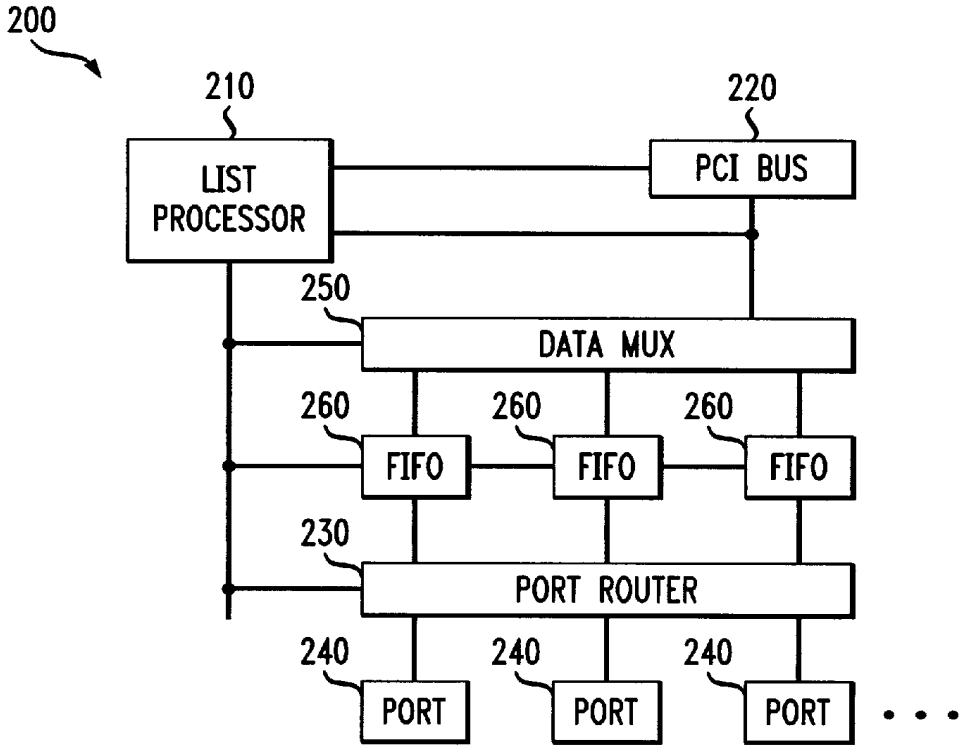


FIG. 3

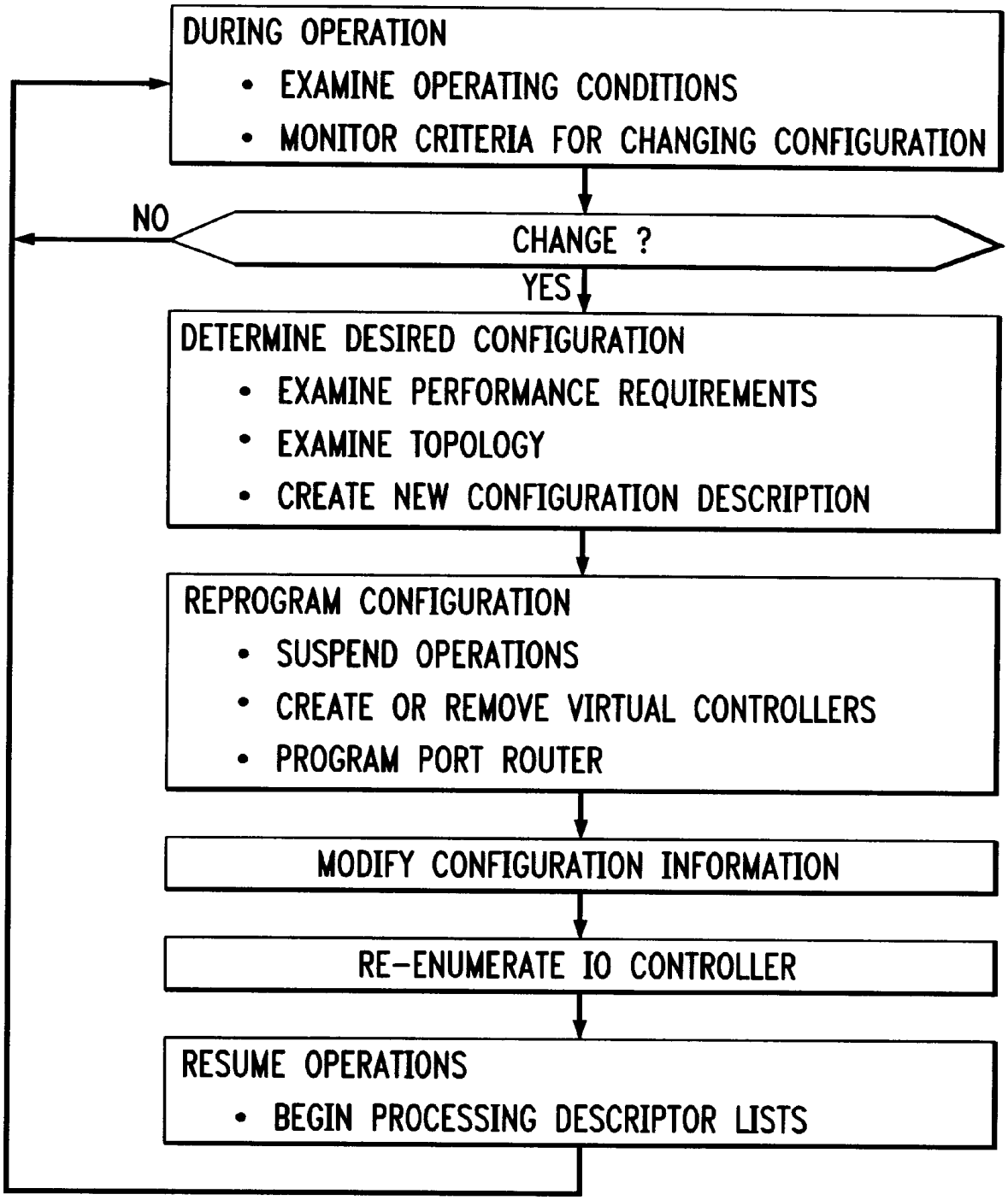


FIG. 4

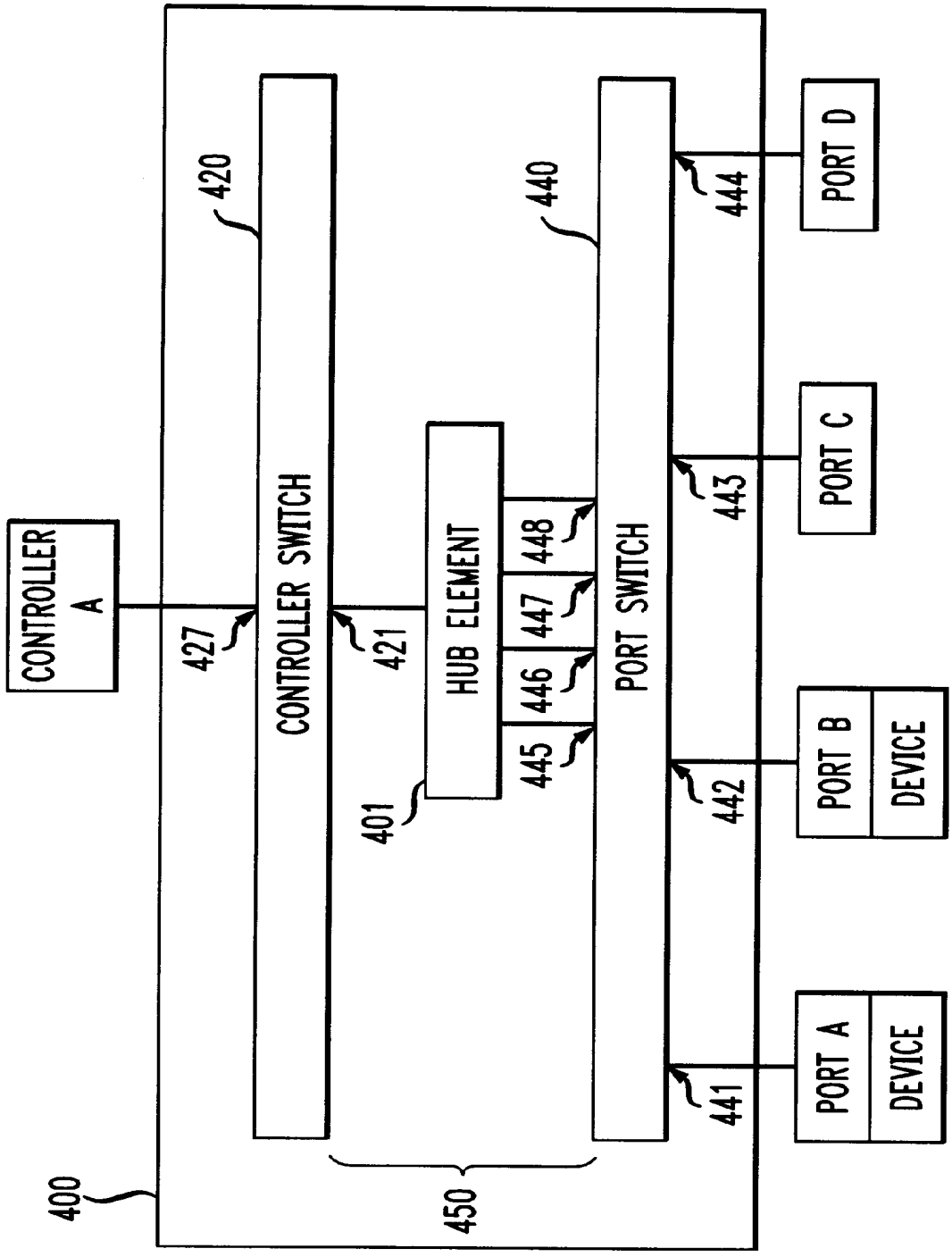
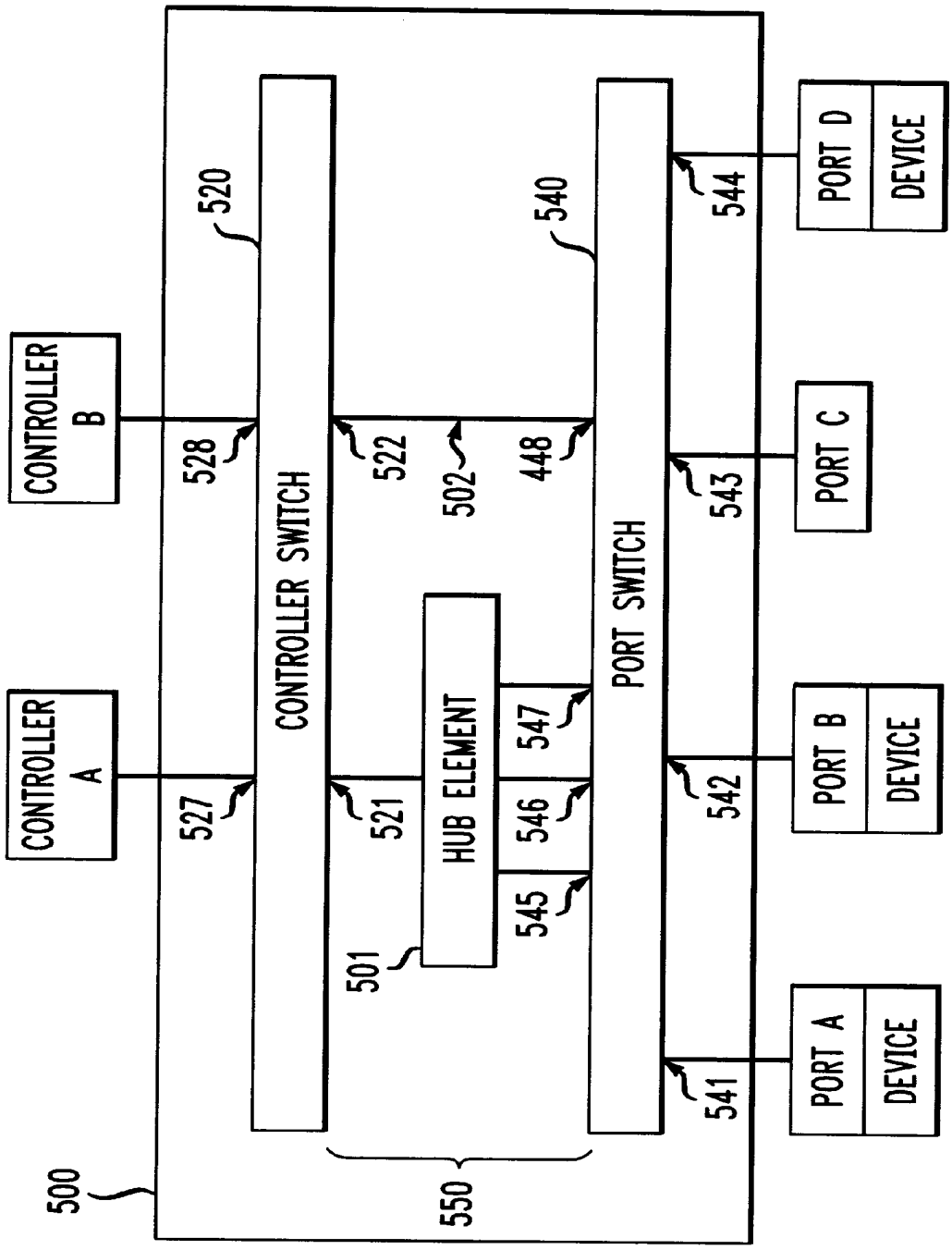


FIG. 5



METHOD AND DEVICE FOR CONTROLLING DATA FLOW THROUGH AN IO CONTROLLER

This application is related to pending U.S. patent application Ser. No. 09/477,593 for "PORT ROUTER" by Aguilar et. al., filed Jan. 4, 2000, the disclosure of which is incorporated herein by reference. This application claims priority to U.S. Provisional Patent Applications Ser. Nos. 60/114,771 filed Jan. 5, 1999, 60/114,772 filed Jan. 5, 1999 and 60/114,767 filed Jan. 6, 1999, which are also incorporated herein by reference.

FIELD OF THE INVENTION

This invention relates to semi-conductor devices and, more particularly, to input/output (IO) controllers.

BACKGROUND OF THE INVENTION

IO controllers provide a connection between a computer operating system and an IO interface. Conventional operating systems create descriptor lists that form the instructions that an IO controller reads and follows in order to do its work. Current implementations of IO controllers include hardware circuits which read and initiate the operations defined in the descriptors. Any significant variation in the descriptor requires a new circuit, consequently rendering existing circuits obsolete. In practice, this creates the situation in which the development of new operating systems and the development of new hardware is delayed because the development of either requires coordination with the other.

The operational speed of an IO controller interface is typically fixed and the controller is limited to the bandwidth of the interface. The addition of another interface requires the addition of an add-on board or the redesign of a motherboard to accommodate a new controller interface device.

Attempts to increase the bandwidth of an IO device have required modification of the device hardware, precluding the dynamic addition of bandwidth. Modification or addition of hardware to a system has numerous disadvantages including reconfiguration expense, additional hardware expense and possible incompatibility with an existing operating system. Further, the addition of new hardware is performed while a system is turned off, requires a technician to install new hardware, and possibly requires a system administrator to change the operating system to support the new hardware. This process can be difficult, error prone and require expensive, time consuming design and re-qualification.

It is therefore desirable to have a device capable of accommodating the variation of a descriptor to add features, improve performance or provide forward or backward compatibility. Additionally, it is desirable to have a device which can be adapted to provide additional bandwidth by dynamically creating new instance of the IO controller in response to the requirements of a system without the addition of new hardware.

SUMMARY OF THE INVENTION

In one aspect of the invention, a method is provided for controlling data flow through an IO controller in a computer system. The method includes determining a desired configuration for the IO controller, reprogramming the IO controller to allow for processing of one or more descriptor lists, modifying the configuration of the IO controller to reflect the addition or deletion of one or more virtual controllers, re-enumerating the IO controller, and processing a descriptor list for each of the IO controller and the one or more virtual controllers. The one or more virtual controllers are discovered and initialized during the re-enumeration and are capable of providing the full bandwidth supported by the IO controller.

In another aspect, the invention is an integrated circuit device for use as an IO controller comprising a system bus interface, a programmable list processor and a port router. The integrated circuit device is adapted for reconfiguration to add or delete one or more virtual controllers. The virtual controllers provide substantially the full bandwidth supported by the integrated circuit device.

The invention may be used in a personal computer system, but can also be applied to other types of compute platforms, including but not limited to information appliances, set-top boxes, cable modems, game consoles, smart appliances, handheld computers, palm-sized computers, embedded control systems, workstations, servers and the like.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of one embodiment of the invention.

FIG. **2** is a block diagram of another embodiment of the invention.

FIG. **3** is a flow diagram illustrating an embodiment of the method of the invention.

FIG. **4** is a block diagram illustrating one embodiment of a port router which can be used with the invention.

FIG. **5** is a block diagram illustrating another embodiment of a port router which can be used with the invention.

DETAILED DESCRIPTION OF THE INVENTION

The invention provides an apparatus and method for dynamically reconfiguring an IO device to accommodate changes in data flow protocol and/or host system interaction. The invention permits the addition (or deletion) of a virtual controller creating a new channel that an operating system can use to transfer data. The IO device can then provide dedicated throughput for peripheral devices that require high bandwidth, and can group several low bandwidth devices to produce to more efficient data transfer. The invention allows for the installation of a new operating program that can accommodate changes in an operating system, changes in system requirements, problems or bugs in devices, and changes in the data flow protocol.

The invention provides a practicable and cost effective solution for integrating additional internal circuitry into a compute platform (i.e. IO controllers). IO controllers create a bridge between an IO stream (bus, interface, network connection) and the operating system drivers in a host computer. The present invention provides a method and apparatus which use programmability to alter the operation of both the IO stream and the operating system interface sides of the bridge.

The invention may be applied to IO controllers including but not limited to USB, IEEE-1394, Fibre Channel, Infiniband and Ethernet. The apparatus and method of the invention can provide increased data flow between peripheral devices and a compute platform. This increased data flow is provided without the need for user intervention and allows for simple installation of complex devices to a compute

platform. The invention may be used in a personal computer system, but can also be applied to other types of compute platforms, including but not limited to information appliances, set-top boxes, cable modems, game consoles, smart appliances, handheld computers, palm-sized computers, embedded control systems, workstations, servers and the like.

An IO controller according to the invention has a longer market life, permits rapid prototyping and implementing of fixes to bugs, accommodates errors in an operating system's descriptor lists, accommodates improvements in descriptor lists, and permits the incorporation of new types of descriptor operation. The invention reduces the number of different products needed to satisfy the requirements of a wide range of systems.

The representations of the FIGURES provided herewith are for purposes of illustration only and is not intended to limit the possible implementations of the invention. The number of ports and IO controllers present in a particular system or application may vary based on system requirements.

In one embodiment of the invention, shown in FIG. **1**, an integrated circuit (IC) device **100** for use as an IO controller is provided having a system bus interface **120**, a programmable list processor **110**, and a port router **130**. The system bus interface **120**, programmable list processor **110**, and port router **130** are operatively connected. The port router **130** is also preferably connected to one or more interface ports **140**.

By employing a programmable list processor **110**, the IO controller can be reprogrammed or reconfigured to process multiple descriptor lists, unlike existing systems which are "single threaded" or capable of executing only a single descriptor list at one time. This functionality allows the IO controller according to the invention to act in the same manner as multiple discrete hardware controllers.

FIG.**4** is a block diagram illustrating one embodiment of a port router **400** which can be used with the invention. The port router **400** comprises a controller switch **420**, a port switch **440** and connection(s) **450** between the controller switch **420** and port switch **440**. Connection(s) **450** comprises a hub element **401**. Port router **400** connects Ports A–D to a single controller, Controller A. The controller switch **420**, the port switch **440** and the connection(s) **450** each have specific input and output capabilities. The port switch **440** has fixed inputs, **441**–**444**, each connected to an interface port. The port switch **440** routes any port to one or more port switch outputs. The port switch **440** has a total number of outputs to support all routing elements. In the example of FIG. **4**, the port switch **440** has four outputs for Hub element **401**. The Hub element(s) preferably comply with the requirements of the IO interface for a given application. Each Hub element combines it's inputs and provides a single output that is input in the controller switch **420**. Some IO interfaces refer to the Hub Element **401** as a bridge, a concentrator or a physical interface. Multiple Hub elements may be used to connect the port switch **440** to controller switch **420**.

The controller switch **420** preferably has one input for each connection to a Hub element and one input for each direct connection (shown in FIG. **2**) to the port switch **440**. The outputs from the controller switch **420** are fixed to match the number of controllers that are necessary or available in a given application. As shown in FIG. **4**, the controller switch has one input **421** connected to Hub element **401**, and one output **427** connected to Controller A. It should be understood that the "inputs" and "outputs" described herein all refer to connections which can handle bi-directional data flow (commonly referred to as "input/outputs")

FIG. **5** illustrates the addition of a new device to Port D of the port router **400** of FIG. **4**, where the bandwidth capacity for Controller A reaches a threshold. Controller B is added to the system and the connection of Port D through Hub element **201** is removed and a direct connection **502** is added. Upon insertion of a device into Port D, the device is detected and it's capacity requirements are reported. This capacity is calculated to be larger than the effective capacity of Controller A. A new controller B is added to the system, and the port router **100** is reconfigured to effect the connection of Ports A, B and C to Controller A and Port D only to Controller B. The connection for Port D to the hub element **501** is removed and a new direct connection **502** is created between the port switch **540** and the controller switch **520**. The connection to Controller B is created from controller switch output **528**. All data flow to and from Port D and the new device flows through this new connection from the port switch output **549** to the controller switch input **522**. All data flow from Ports A, B and C continues to have the same routing from port switch outputs **545**, **546** and **547** to hub element **501**. In this configuration the data flow from Ports A, B and C are combined and delivered to the controller switch input **521**.

In operation, the port switch **400** will preferably start in a default state with certain port inputs connected to a desired Controller. The example of FIG. **4** shows a system having a default state with four Ports, two devices connected to Ports A and B, one Hub Element **401**, and one Controller A. Through a set of hardware registers (not shown) preferably resident in the port router **400**, software operating in the system can query the port router **400** and discover the default topology. The system software can effect a change in routing of signals between any combination of Ports and Controllers by programming values in the hardware registers. During operation an event occurs. An event may be a new device being removed or inserted into a Port. This event triggers the software to examine the requirements of the new device. These requirements may include the device's data capacity requirements (bandwidth and latency) and its data style (asynchronous, isochronous, burst, stream). The new requirements are combined with the current set of requirements from devices already installed, and a new preferably optimized topology for routing the ports is computed. Preferably, the system software causes IO operations to suspend and effect the re-routing between the Ports and IO Controller(s). The system software then resumes IO operations and the optimized routing becomes the new routing until a new event occurs.

In another embodiment the port router **400** preferably contains embedded software which allows routing and re-routing to be effected internally without requiring the intervention of system software. This provides the ability for self-monitoring and dynamic load balancing based on the flow of data through the port router **400**.

In another embodiment of the invention, shown in FIG. **2**, an integrated circuit (IC) device **200** for use as an IO controller is provided having a peripheral component interconnect (PCI) bus **220** as the system bus interface, a programmable list processor **210**, a port router **230**, a data mux **250**, and a plurality of FIFO(first in first out) data buffers **260**. One or more interface ports **240** are also preferably connected to the port router **230**. Each of the elements are preferably operatively connected to allow data signals and control signals to be transmitted between the elements.

When data is received from the interface it preferably passes through a Port **240** and the Port Router **230** preferably directs it to a particular FIFO **260**. When a data element is received in the FIFO **260**, the List Processor **210** preferably examines it and determines its disposition. The actions taken on the data element depend on the protocol rules for the interface and will be different for example with Ethernet, USB, IEEE-1394 and the like. It is preferable that controlling software in the List Processor **210** is responsible for determining a desired configuration. The List Processor **210** preferably performs an action such as but not limited to: transmit the data element to system memory, cause a pending transaction to be completed, cause an error to be transmitted to the system, cause an error to be transmitted to the interface, prepare data for transmission to the interface. All data elements transmitted to and from system memory through the PCIbus **220** preferably pass through the Data Mux **250** element. When a data element is to be transmitted from system memory to the interface, it is preferably received through the PCIbus **220**, passed through the data mux **250** and placed into the correct FIFO **260**. The List Processor that initiated fetching the data from system memory preferably prepares it for transmission to the interface and monitors the conditions on the interface for the appropriate opportunity for data transmission. If the interface is non responsive or not operational the List Processor **210** preferably may take several actions: save data for later retransmission, report an error to the system, and the like.

The List Processor element **210** preferably has many characteristics of a general purpose compute engine such as a microprocessor or microcontroller with adaptations to perform the functions of controlling the flow of data between the Port and system bus. It executes software that preferably is fixed in ROM or that more preferably may be downloaded. The software is preferably written to reflect the manner in which the protocol on the interface is defined, typically following industry standards. It is preferred that the software can also be applied for private interface standards. If for example, the protocol requires that the system transmits a data element and receives an acknowledgment that the data was correctly received, the List Processor element has a software control structure that anticipates the receipt of the acknowledgment, if no acknowledgment is received then perhaps a re-transmission is initiated or an error is transmitted to the system. The exact action depends on the rules of the protocol definition.

The system bus **120** may also comprise other multi-master bus structures including but not limited to PCI-X, Infiniband, VMNEbus, HubLink, and the like. A common feature of these operating systems is an enumeration phase. During enumeration, the hardware present on a system bus is queried, the data describing the features of the hardware is created and acted upon by operating system software. The operating system used with the invention is preferably capable of performing re-enumeration if requested.

It is preferred that the invention has a default configuration that is reported during enumeration. In a preferred default state, the list processor **110** is programmed to process a single list and is configured to report a single **10** controller to the operating system. It is also preferred that the default state on start up is the state maintained from before the previous power down.

During operation the list processor **110** preferably functions to interpret and act upon the instructions of an operating system. A virtual controller may be created to allow the operating system to understand and act as if multiple instances of discrete IO controller hardware exist.

Modifying the hardware description of the IO controller and requesting that the operating system re-enumerate the IO controller allows the creation or deletion of one or more new virtual instance of the IO controller. The dynamic creation of a new IO controller can be performed multiple times, until practical limits (i.e. system bus bandwidth, IO controller number limits, etc) are reached.

In one embodiment of the method of the invention illustrated in the flow diagram of FIG. **3**, a determination is made that the current IO controller configuration should be modified. The determination may be made by the operating system or by a monitoring element (not shown) internal to the IO controller. The determination that the current IO controller configuration should be modified is preferably made based on the existence of one or more predetermined conditions including but not limited to increased or decreased data flow through the IO controller.

The determination is preferably based on a number of criteria including but not limited to: the total bandwidth requirements of peripheral devices, the bandwidth requirements of a particular peripheral, the latency requirements of a particular peripheral, the presence of conflicting peripherals, the topology of the peripherals, the addition or removal of a peripheral, the increase or decrease in the bandwidth requirements of an existing peripheral, the increase or decrease in the latency requirements of an existing peripheral, and the such.

The operation of the IO controller is then preferably temporarily suspended. The devices attached to the IO controller are suspended (or put to sleep) to stop the data flow. The list processor **110** is then preferably programmed to allow the processing of one or more new descriptor lists for each virtual controller added to the system. The descriptor list(s) previously in operation remain intact and are processed along with the new descriptor list(s) when operation resumes. The configuration of the IO controller is modified to reflect the addition or deletion of one or more virtual controllers. Preferably, the devices connected to the IO controller are re-routed to a desired configuration which provides efficient data paths thus optimizing or improving the data flow through the IO controller. This is typically accomplished by dividing the data flow between the one or more IO controllers.

After reconfiguration is complete, a request is made to the operating system to re-enumerate the IO controller. The virtual controllers are detected, and the operating system queries to discover which devices are connected to which controllers. Descriptor lists are created and the list processor **110** begins to process each descriptor list. The devices may then resume operation. Typically the devices are re-enumerated by the operating system.

Features of the list processor **110** include multi-threaded execution in which multiple descriptor lists may be operated on simultaneously. Typical operations which can be executed simultaneously by the list processor **110** include: fetching from system memory, caching (or fetching descriptors before they are needed to reduce multiple accesses to system memory), validation of the descriptor, preparation of IO hardware for data transfer, construction of data flow protocol elements, fetching or delivering of data from and to system memory, monitoring of data flow for exceptions, monitoring of data flow for completion, termination of descriptor, error processing and cleanup, and reporting results back to the operating system.

Thus, the method of controlling data flow through an IO controller in a computer system according to an embodiment

of the invention comprises determining a desired configuration for the IO controller, reprogramming the IO controller to allow for processing of one or more descriptor lists, modifying the configuration of the IO controller to reflect the addition or deletion of one or more virtual controllers, re-enumerating the IO controller, and processing a descriptor list for each of the IO controller and the one or more virtual controllers. The one or more virtual controllers are preferably discovered and initialized during the reenumeration and preferably provide the full bandwidth supported by the IO controller and defined by the interface specification.

Modifying the configuration of the IO controller preferably includes creating one or more new configuration descriptors which indicate the presence of the one or more virtual controllers. Connections from the interface ports are preferably rerouted to a desired routing configuration which provides efficient data flow. Determining a desired configuration for the IO controller and modifying the configuration of the IO controller is preferably controlled by drivers in the computer system, or more preferably by firmware embedded in the IO controller. Determining a desired configuration for the IO controller preferably includes determining whether the existing configuration of the IO controller should be changed based on existing conditions and determining an optimized configuration for the IO controller based on the existing conditions.

The method of controlling data flow through an IO controller in a computer system according to another embodiment of the invention comprises entering a reconfiguration mode upon the existence of one or more predetermined conditions, determining a desired configuration for the IO controller, reprogramming the IO controller to allow for processing of one or more descriptor lists, modifying the configuration of the IO controller to reflect the addition or deletion of one or more virtual controllers, re-enumerating the IO controller, and processing a descriptor list for each of the IO controller and the one or more virtual controllers. Entering a reconfiguration mode preferably includes suspending operation of devices connected to the computer system. The predetermined conditions preferably include increased or decreased data flow through the IO controller.

In another aspect, the invention is an integrated circuit device for use as an IO controller comprising a system bus interface, a programmable list processor and a port router. The system bus interface, the programmable list processor and the port router are preferably operatively connected. The integrated circuit device is preferably adapted for reconfiguration to add or delete one or more virtual controllers, the virtual controllers providing substantially the full bandwidth supported by the integrated circuit device.

Although the invention has been described with reference to exemplary embodiments and accompanying drawings, it can be readily understood that the invention is not restricted to such embodiments and that various changes and modifications can be made by those skilled in the art without departing from the spirit and scope of the invention.

What is claimed is:

1. A method of controlling data flow through an IO controller from one or more port interfaces to internal circuitry of a computer system comprising:

determining a desired configuration for said IO controller;

reprogramming said IO controller to allow for processing of one or more descriptor lists;

modifying the configuration of said IO controller to reflect the addition or deletion of one or more virtual controllers;

re-enumerating said IO controller;

processing a descriptor list for each of said IO controller and said one or more virtual controllers; and

wherein port interfaces support a maximum bandwidth, and said one or more virtual controllers are discovered and initialized during said re-enumeration and provide the maximum bandwidth supported by said port interfaces.

2. A method according to claim **1**, wherein modifying the configuration of said IO controller includes creating one or more new configuration descriptors which indicate the presence of said one or more virtual controllers.

3. A method according to claim **1**, further comprising re-routing connections to said IO controller to a desired routing configuration.

4. A method according to claim **1**, wherein determining a desired configuration for said IO controller and modifying the configuration of said IO controller is controlled by drivers in said computer system.

5. A method according to claim **1**, wherein determining a desired configuration for said IO controller and modifying the configuration of said IO controller is controlled by firmware embedded in said IO controller.

6. A method according to claim **1**, wherein determining a desired configuration for said IO controller includes determining whether an existing configuration of said IO controller should be changed based on existing conditions and determining an optimized configuration for said IO controller based on said existing conditions.

7. A method of controlling data flow through an IO controller in a computer system comprising:

entering a reconfiguration mode upon the existence of one or more predetermined conditions;

determining a desired configuration for said IO controller;

reprogramming said IO controller to allow for processing of one or more descriptor lists;

modifying the configuration of said IO controller to reflect the addition or deletion of one or more virtual controllers;

re-enumerating said IO controller;

creating a descriptor list for each of said IO controller and said one or more virtual controllers; and

wherein said predetermined conditions include increased or decreased data flow through said IO controller.

8. A method according to claim **7**, wherein entering a reconfiguration mode includes suspending operation of devices connected to said computer system.

9. An integrated circuit device for use as an IO controller comprising:

a system bus interface;

a programmable list processor; and

a port router;

wherein said system bus interface, said programmable list processor and said port router are operatively connected;

wherein said IO controller processes one or more descriptor lists;

said IO controller reflects the addition or deletion of one or more virtual controllers;

said IO controller is re-enumerated;

a descriptor list is processed for each of said IO controllers and said one or more virtual controllers; and

wherein port interfaces support a maximum bandwidth, and said one or more virtual controllers are discovered

and initialized during said re-enumeration and provide the maximum bandwidth supported by said port interfaces.

10. A method of controlling data flow through an IO controller from one or more port interfaces to internal circuitry of a computer system comprising:

determining a desired configuration for said IO controller;

reprogramming said IO controller to allow for processing of one or more descriptor lists;

modifying the configuration of said IO controller to reflect the addition or deletion of one or more virtual controllers;

re-enumerating said IO controller;

processing a descriptor list for each of said IO controller and said one or more virtual controllers; and

wherein one or more port interfaces support a maximum bandwidth, and said one or more virtual controllers are discovered during said re-enumeration and provide the maximum bandwidth supported by said one or more port interfaces.

11. A method according to claim **1**, wherein said one or more virtual controllers are initialized during said re-enumeration after being discovered.

12. An integrated circuit device for use as an IO controller comprising:

a system bus interface;

a programmable list processor; and

a port router;

wherein said system bus interface, said programmable list processor and said port router are operatively connected;

wherein said IO controller processes one or more descriptor lists;

said IO controller reflects the addition or deletion of one or more virtual controllers;

said IO controller is re-enumerated;

a descriptor list is processed for each of said IO controllers and said one or more virtual controllers; and

wherein one or more port interfaces support a maximum bandwidth, and said one or more virtual controllers are discovered during said re-enumeration and provide the maximum bandwidth supported by said one or more port interfaces.

* * * * *

Delaware

The First State

I, HARRIET SMITH WINDSOR, SECRETARY OF STATE OF THE STATE OF DELAWARE, DO HEREBY CERTIFY THE ATTACHED IS A TRUE AND CORRECT COPY OF THE CERTIFICATE OF OWNERSHIP, WHICH MERGES:

"LSI SUBSIDIARY CORP.", A DELAWARE CORPORATION,

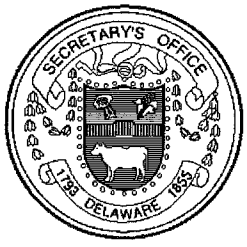
WITH AND INTO "LSI LOGIC CORPORATION" UNDER THE NAME OF "LSI CORPORATION", A CORPORATION ORGANIZED AND EXISTING UNDER THE LAWS OF THE STATE OF DELAWARE, AS RECEIVED AND FILED IN THIS OFFICE THE FIFTH DAY OF APRIL, A.D. 2007, AT 8:09 O'CLOCK A.M.

AND I DO HEREBY FURTHER CERTIFY THAT THE EFFECTIVE DATE OF THE AFORESAID CERTIFICATE OF OWNERSHIP IS THE SIXTH DAY OF APRIL, A.D. 2007.

A FILED COPY OF THIS CERTIFICATE HAS BEEN FORWARDED TO THE NEW CASTLE COUNTY RECORDER OF DEEDS.

2109844 8100M

070402663



Harriet Smith Windsor

Harriet Smith Windsor, Secretary of State

AUTHENTICATION: 5568399

DATE: 04-05-07

State of Delaware
Secretary of State
Division of Corporations
Delivered 08:12 AM 04/05/2007
FILED 08:09 AM 04/05/2007
SRV 070402663 - 2109844 FILE

CERTIFICATE OF OWNERSHIP AND MERGER

MERGING

LSI SUBSIDIARY CORP.

WITH AND INTO

LSI LOGIC CORPORATION

Pursuant to Section 253 of the General Corporation Law of the State of Delaware

LSI Logic Corporation, a Delaware corporation ("LSI Logic" or the "Corporation"), HEREBY CERTIFIES AS FOLLOWS;

FIRST: LSI Logic is a corporation incorporated on December 5, 1986 pursuant to the General Corporation Law of the State of Delaware.

SECOND: LSI Logic owns all of the outstanding shares of capital stock of LSI Subsidiary Corp., a corporation incorporated on March 26, 2007 pursuant to the General Corporation Law of the State of Delaware ("Subsidiary").

THIRD: LSI Logic, by the following resolutions of its Board of Directors, duly adopted at a meeting on April 2, 2007 and filed with the minutes of its Board of Directors, determined to merge Subsidiary with and into LSI Logic, and LSI Logic does hereby merge Subsidiary with and into LSI Logic effective as of the Effective Time (as defined below):

WHEREAS, LSI Logic owns all of the outstanding shares of capital stock of Subsidiary;

WHEREAS, LSI Logic desires, on behalf of itself and in its capacity as the sole stockholder of Subsidiary, to merge Subsidiary with and into LSI Logic pursuant to the provisions of Section 253 of the Delaware General Corporation Law; and

WHEREAS, it is intended that the merger of Subsidiary with and into LSI Logic will constitute a liquidation under Section 332 of the Internal Revenue Code and/or a reorganization under Section 368(a) of the Internal Revenue Code.

NOW, THEREFORE, BE IT RESOLVED, that Subsidiary merge (the "Merger") with and into the Corporation;

RESOLVED, that the Merger shall become effective on April 6, 2007 (the "Effective Time") upon the filing of a Certificate of Ownership and Merger with

the Secretary of State of the State of Delaware in accordance with the provisions of the Delaware General Corporation Law;

RESOLVED, that, at the Effective Time, Subsidiary shall be merged with and into the Corporation, the separate existence of Subsidiary shall cease, and the Corporation shall continue as the surviving corporation of the Merger, and the Corporation, without further action, shall possess all the properties, rights, privileges, powers and franchises, public and private, of both the Corporation and Subsidiary, and shall be subject to all debts, liabilities, obligations, restrictions, disabilities and duties of both the Corporation and Subsidiary;

RESOLVED, that the Restated Certificate of Incorporation of the Corporation, as in effect immediately prior to the Effective Time, shall remain the certificate of incorporation of the Corporation from and after the Effective Time, without change, until thereafter amended as provided by law or such certificate of incorporation; provided, however, that, effective as of the Effective Time, the name of the Corporation shall be changed from "LSI Logic Corporation" to "LSI Corporation" and Article I of the Restated Certificate of Incorporation of the Corporation shall be amended to read in its entirety as follows:

"1. The name of the corporation is LSI Corporation (the "Corporation")."

RESOLVED, that the by-laws of the Corporation, as in effect immediately prior to the Effective Time, shall remain the by-laws of the Corporation from and after the Effective Time, without change, until thereafter amended as provided by law, the certificate of incorporation of the Corporation or such by-laws;

RESOLVED, that the directors of the Corporation immediately prior to the Effective Time shall remain the directors of the Corporation from and after the Effective Time, without change, each to hold office in accordance with the certificate of incorporation and by-laws of the Corporation until their successors are duly elected or appointed and qualified or until their earlier, death, resignation or removal;

RESOLVED, that the officers of the Corporation immediately prior to the Effective Time shall remain the officers of the Corporation from and after the Effective Time, without change, each to hold office in accordance with the certificate of incorporation and by-laws of the Corporation until their successors are duly elected or appointed and qualified or until their earlier, death, resignation or removal;

RESOLVED, that, at the Effective Time, each issued and outstanding share of the Common Stock, par value $0.01 per share, of Subsidiary ("Subsidiary Common Stock") held by the Corporation shall, without any action on the part of the Corporation or Subsidiary, be canceled without any conversion thereof or any consideration therefore and no payment or distribution shall be made with respect

thereto, and each issued and outstanding share of the Common Stock, par value $0.01 per share, of the Corporation shall remain outstanding following the Effective Time without change;

RESOLVED, that officers of the Corporation be, and each of them acting alone hereby is, authorized to make, execute and file with the Secretary of State of the State of Delaware a Certificate of Ownership and Merger setting forth a copy of these resolutions providing for the Merger of Subsidiary with and into the Corporation and the Corporation's assumption of Subsidiary's obligations and the date of adoption thereof; and

RESOLVED, that officers of the Corporation be, and each of them acting alone hereby is, authorized to take all other actions and to prepare, execute, deliver and file all other agreements, instruments, documents and certificates in the name and on behalf of the Corporation and to pay all such fees and expenses as they, or any one of them, may deem necessary, proper or advisable in order to effect the Merger, and that any actions of any officer of the Corporation authorized by the foregoing resolutions or that would have been authorized by any of the foregoing resolutions except such actions were taken prior to the adoption of these resolutions be, and they hereby are, ratified, approved and confirmed as actions of the Corporation.

FOURTH: That anything herein or elsewhere to the contrary notwithstanding, the Merger may be amended or terminated and abandoned by the Board of Directors of LSI Logic at any time prior to the time that the Merger becomes effective.

IN WITNESS WHEREOF, LSI Logic has caused this Certificate of Ownership and Merger to be signed by a duly authorized officer, and attested by its Corporate Secretary, this 4th day of April, 2007.

By: Bryon Look
Name: Bryon Look
Title: Executive Vice President and Chief Financial Officer

ATTEST:

By: Jean Rankin
Name: JEAN F. RANKIN
Title: EXECUTIVE VICE PRESIDENT and General Counsel